Universidad Zaragoza
1542

Proyecto Fin de Carrera

Desarrollo de un termómetro inalámbrico basado en ISO/IEEE 11073

Autor:
Sergio Beltrán Nuez

Directores:
Antonio Aragües Ruiz
Álvaro Marco Marco

Escuela de ingeniería y arquitectura
Ingeniería de telecomunicaciones

Septiembre 2013

AGRADECIMIENTOS

Me gustaría agradecer a Álvaro y Antonio así como al resto de compañeros su ayuda y disponibilidad.

También me gustaría agradecer especialmente a mis padres y hermanos que nunca me haya faltado su apoyo. Y como no a Carlota por su paciencia ilimitada.

RESUMEN DEL PROYECTO FIN DE CARRERA

"**Desarrollo de un termómetro inalámbrico basado en ISO/IEEE 11073**"

Realizado por: Sergio Beltrán Nuez **Dirigido por:** Álvaro Marco

Una situación muy usual que podemos encontrar en un hospital es el hecho de que un paciente este sujeto a una monitorización permanente de medidas como la presión arterial, la saturación del oxígeno en sangre o la temperatura. Todas estas medidas van conectadas a una interfaz de monitorización por medio de cables, lo que conlleva una serie de problemas relacionados con la movilidad del paciente, tanto para el propio paciente como para el personal sanitario.

Este Proyecto Fin de Carrera (PFC) plantea construir un entorno de telemonitorización inalámbrica utilizando tecnología ZigBee y con el fin de trabajar de acuerdo a los estándares del Comité Europeo de Normalización (CEN) implantar el estándar ISO/IEEE 11073. Para establecer la comunicación entre un manager y un agente médico.

Este proyecto se apoya en dos tecnologías como son ZigBee y X73, la parte correspondiente a ZigBee se va a desarrollar en el laboratorio del grupo Howlab, un grupo de investigación de la universidad de Zaragoza y contando con el asesoramiento del profesor Álvaro Marco. Por otro lado en lo que corresponde a la implantación del estándar médico se va a llevar a cabo dentro de la empresa Goodday Solutions S.L, trabajando bajo la supervisión del director tecnológico Antonio Aragües.

El PFC comenzará con el estudio de la tecnología ZigBee, para conseguir un software que de un modo automático cree y gestione una red ZigBee, además de llevar a cabo una configuración específica de los módulos que la componen. Por otro lado implementaremos el perfil médico ZigBee Health Care (ZHC) a ambos extremos de la comunicación el cual nos permite establecer un canal sobre el que establecer el protocolo de comunicación ISO/IEEE 11073, que será el encargado de realizar los pasos necesarios para acabar estableciendo una conexión entre un agente y un manager de un modo seguro, con el fin de transmitir un dato clínico.

Para gestionar correctamente cada nivel de la comunicación también será necesario el encapsulado y desencapsulado de los datos dentro de tramas ZCL, ZHC y APDU en ambos extremos.

Para llevar a cabo el proyecto se dispondrá de dos dispositivos ZigBee de Telegesis, uno de estos módulos haremos que se comporte como el manager, mientras que el otro módulo junto con el sensor emularán el comportamiento del dispositivo médico.

El proyecto se desarrollara en un entorno Java y siguiendo la especificación OSGi, para lo que se utilizará como entorno de ejecución Karaf (http://karaf.apache.org), donde también se implementará una interfaz gráfica para poder seguir todo el proceso de comunicación.

De manera general, este PFC busca la implementación de una arquitectura y un modo sistemático de trabajo para poder extrapolarlo en un futuro a una monitorización completa del paciente con todo tipo de sensores en un entorno ZigBee.

Índice de contenidos

Índice de Figuras

Índice de tablas

Acrónimos y siglas

AARE	A-Aassociate Response
AARQ	A-Association Request
ACK	Acknowledgement
AENOR	Asociación Española de Normalización y Certificación
ANSI	*American National Standards Institute*
APDU	Application Protocol Data Unit
APS	American Physical Society
ATA	American Telemedicine Asociation
CBA	Commercial Building Automation
CEN	Comité Europeo de Normalización
COO	Coordinador
CSMA/CA	Carrier Sense Multiple Access/Collision Avoidance
EUI	Extended Unique Identifier
FFD	Dispositivo de funcionalidad completa
GOF	Trama general de operaciones
HA	Home Automation
HL7	Health Level 7
ID	Identificador
IEEE	Institute of Electrical and Electronics Engineers
ISCIII	Instituto de Salud Carlos tercero
ISO	International Organization for Standardization
MAC	Control de Acceso al Medio
MDER	Medical Device Encoding Rules
NHS	Medical Device Encoding Rules
NWK	Capa de red
OSGi	Open Services Gateway Initiative
PAN ID	Identificador de red
PC	Personal computer
PFC	Proyecto fin de carrera
PHD	Personal Health Data
PHY	Capa física
PoC	Point-OfCare
POM	*Project Object Model*

RF4CE	Radio Frequency for Consumer. Electronics
PRST	Presentation Apdu
RLRE	Release Response
RLRQ	Release Request
SE	Smart Energy
SED	Sleepy End Device
TA	Telecom Applications
TIC	Tecnologías de la información y las comunicaciones
UPNA	Universidad pública de Navarra
USB	Universal Serial Bus
UZ	Universidad de Zaragoza
WiFi	Wireless Fidelity
ZCL	ZigBee Cluster Library
ZDO	ZigBee Device Object
ZDP	ZigBee Device Profile
ZHC	ZigBee Health Care
ZIP	ZigBee IP

1 Introducción y objetivos

1.1 Introducción

Cada vez más las tecnologías inalámbricas están más presentes en nuestra sociedad y resulta evidente que con el paso del tiempo han ido adoptando una manera cada vez más sencilla de utilizar cualquier tipo de dispositivo con el fin de hacernos la vida más cómoda. Es por tanto obvio que su implantación en la sociedad ha venido para quedarse y que cada vez son más los entornos donde encontramos este tipo de tecnologías.

Un campo que se ha visto impulsado en los últimos años es el campo de la medicina, más concretamente el ámbito de la e-salud que se trata de la combinación de las tecnologías de la información y las comunicaciones (TIC) con los cuidados médicos y cubre desde el diagnóstico hasta el seguimiento de los pacientes, pasando por la gestión de las organizaciones implicadas en estas actividades.

Es en este campo de la e-salud donde se va a centrar este proyecto, se va a desarrollar la creación de un entorno de monitorización inalámbrica compuesto por un agente o dispositivo médico, que en nuestro caso será un termómetro, y un manager. Dicha comunicación tendrá que implementarse de acuerdo a los estándares médicos para que sea capaz de operar en un entorno real.

Para cubrir la parte de la comunicación inalámbrica se va a utilizar tecnología ZigBee, que se trata de una tecnología relativamente nueva, ya que todavía se encuentra en una fase temprana de su implantación, pero resulta idónea para este tipo de comunicación ya que ha sido diseñada específicamente para trabajar con entornos donde se requieran comunicaciones seguras con baja tasa de envío de datos y maximización de la vida útil de las baterías en los dispositivos.

Por otro lado, a la hora de trabajar con dispositivos médicos existe una gran problemática en lo referente a la interoperabilidad entre los protocolos de diferentes fabricantes ya que el formato de comunicación que utilizan los dispositivos es todavía propio de cada fabricante y esto repercute a la hora de manejar los datos. Por eso uno de los grandes retos para los próximos años es implantar una estandarización en los dispositivos de e-salud con el fin de conseguir la interoperabilidad de equipos diseñados por diferentes fabricantes.

Para solucionar este problema se ha pensado en la utilización de estándar de interoperabilidad ISO/IEEE 11073-Personal Health Devices [1], el cual cuenta con el apoyo de organismos de estandarización nacional (AENOR) e internacional (ISO, IEEE), además de estar apoyado por Continua Health Alliance [15].

1.2 Antecedentes

Este proyecto nace de la combinación de dos grupos de investigación de la universidad de Zaragoza. Por un lado tenemos Howlab [2], que forma parte del Instituto de Investigación en Ingeniería de Aragón y trabaja en el desarrollo y diseño de proyectos de hardware y software libre y por otro lado esta Goodday Solutions S.L. [3], una empresa especializada en la integración de soluciones en dispositivos médicos. Ambos grupos presentan una gran innovación en el campo I+D buscando soluciones innovadoras ante problemas cotidianos.

A lo largo del desarrollo del proyecto cada uno de los grupos aportará sus conocimientos y experiencia en el campo en el que son expertos, por un lado Howlab nos proporciona su respaldo respecto a la tecnología ZigBee. De hecho se utilizará como base para desarrollar este PFC, un proyecto código abierto llamado Zeta Project [29] que ha sido desarrollado por Howlab y viene apoyado en un entorno OSGi (OSGi se trata de un sistema modular para Java que establece las formas de crear módulos y de que estos interactúen entre sí en tiempo de ejecución [33]). El Zeta Project nos proporciona una serie de herramientas necesarias para poder gestionar nuestra red ZigBee, y al mismo tiempo poder implementar el envío de mensajes y la configuración de los dispositivos ZigBee, para esta parte del proyecto se contará con el asesoramiento y la supervisión del profesor de la universidad de Zaragoza Álvaro Marco.

Por otro lado, Goodday Solutions proporciona el soporte necesario para gestionar el protocolo X73 y adoptando de este modo la conectividad del dispositivo, en esta parte se trabajará bajo la supervisión del director tecnológico Antonio Aragüés. Para completar esta fase de la implementación del protocolo 11073 se utiliza como base el trabajo desarrollado por otro proyecto fin de carrera en la universidad de Zaragoza, este proyecto a su vez partió del trabajo de una plataforma desarrollada por el grupo X73Spain [4], que está formado por la Universidad de Zaragoza (UZ), la Universidad Politécnica de Navarra (UPNA) y el Instituto de Salud Carlos III (ISCIII), este proyecto lleva el nombre de *"Optimización de una Plataforma Telemática para Monitorización de Pacientes orientada a u-Salud, y basada en Estándares y Plug-and-Play"*

De este modo, debido a la diferencia que existe entre las tecnologías utilizadas para llevar a cabo el proyecto y teniendo en cuenta que el desarrollo del proyecto se dividen en bloques equitativos, se ha decidido que la mejor manera de llevarlo a cabo sea mediante la co-dirección del proyecto, centrándose cada tutor en la tecnología en la que es experto.

1.3 Motivación

Este Proyecto Fin de Carrera se inicia con la motivación de tratar de dar una pequeña aportación a un sistema innovador que se está abriendo paso en un mercado muy competitivo como es el mercado hospitalario, además a esto tenemos que sumar el hecho de trabajar en un sistema de código abierto y la posibilidad de trabajar con una tecnología nueva para mí.

Si analizamos el aspecto tecnológico del proyecto y porque se ha decidido utilizar este tipo de tecnología, en cuanto a la tecnología inalámbrica se ha

elegido ZigBee ya que resulta idónea para este tipo de comunicación ofreciéndonos una serie de ventajas que no nos pueden ofrecer otras tecnologías como son Bluetooth o Wi-Fi. Por otro lado con la finalidad de aportarle la conectividad necesaria para trabajar en un entorno médico, se optó por implantar el protocolo X73, que se trata de un estándar adoptado por todos los países europeos para la interoperabilidad de dispositivos médicos. Aportando una conectividad completa: transparencia, Plug&Play, y facilidad de uso. Al resultar un estándar compatible con ZigBee resulta el complemento perfecto en la recogida de datos clínicos.

Por otro lado si trasladamos el análisis a un plano más personal, a la hora de decantarme por este proyecto buscaba un proyecto centrado en la parte de comunicaciones, pero que a su vez quedara abierto a otros campos. Además de esto también buscaba que me ofreciera la posibilidad de introducirme en una nueva materia y poder formarme con nuevos conocimientos, de hecho el desarrollar el proyecto en un entorno de trabajo diferente a lo utilizado hasta ahora, además de abrirme la puerta de nuevas tecnologías me ha permitido aprender a desenvolverme en un entorno de desarrollo Java.

Por lo tanto se puede afirmar que se trata de un proyecto que partiendo de una serie de trabajos e investigaciones anteriores, procede a combinar todos estos aspectos formando así una arquitectura ordenada que acabe dando lugar a una plataforma de trabajo y que en un momento dado, pueda llevarse a una futura explotación, o servir como base para futuros proyectos.

1.4 Objetivos

El objetivo del proyecto es desarrollar un sistema capaz de dotar de conectividad e interoperabilidad a un manager y un agente médico, a través del protocolo 11073 utilizando una conexión inalámbrica ZigBee. Con el fin de que pueda operar en un entorno médico real, será necesario respetar todas las normas y los estándares necesarios. Todo ello se va a desarrollar en una plataforma de código abierto y prestando especial atención a los aspectos de modularidad y conectividad.

Si realizamos un análisis global de lo que se pretende conseguir una vez finalizado el proyecto tenemos por un lado un dispositivo médico capaz de tomar datos de temperatura y transmitírselos a través de la comunicación inalámbrica ZigBee a un Manager, mediante una serie de normas de comunicación que viene marcadas por el estándar 11073 [1].

Todo esto se llevará a cabo de un modo automático mediante el desarrollo de dos módulos en Java, encargados de gestionar la parte correspondiente al manager y al agente respectivamente. Para gestionar ambos extremos de la comunicación se requiere la existencia de un PC en cada uno de los extremos del canal, aunque en nuestro caso se desarrollará a partir de dos instancias diferentes dentro del mismo ordenador.

En cuanto al material utilizado, para implementar el manager se utilizará un dispositivo ZigBee de la empresa Telegesis [5], que además de realizar gestiones a nivel de red nos permite configurar el dispositivo a través de un sistema de registros. Mientras que para diseñar el agente ya que no disponemos de un dispositivo médico real, lo emularemos a través de otro módulo como el utilizado en el manager que tomará la medida de temperatura de un sensor externo, en nuestro caso el sensor de temperatura se tratará de

un módulo ZigBee, aunque no tendría por qué tratarse obligatoriamente de un dispositivo ZigBee ya que su única función es transmitirle el dato de temperatura.

A lo largo del proyecto nos vamos a centrar en establecer la comunicación entre el manager y el agente, no obstante en el anexo 4 se puede observar como se ha desarrollado el bloque que emula el agente, en la Figura 1.1 encontramos un análisis gráfico de la estructura final que tendrá el proyecto, donde se muestra la parte del manager y la correspondiente a la emulación del agente.

El proyecto se compondrá de tres fases, en primer lugar la parte correspondiente a la tecnología ZigBee, que comprenderá un estudio detallado de la tecnología para poder llevar a cabo la gestión y creación de la red, aportando la comunicación necesaria entre todos los dispositivos a través de la red y además se realiza un estudio de la configuración necesaria en cada dispositivo. Esta configuración comprenderá por un lado la configuración de los registros para que cada uno cumpla su función dentro de la red y por otro lado para que sea capaz de soportar el perfil público que nos proporciona ZigBee Alliance [6] para tratar con dispositivos médicos como es el ZigBee Health Care [7].

La implantación y el estudio de este perfil conformarán el segundo bloque del PFC, una vez conectado el manager con el agente se pasará a establecer entre ellos un canal de comunicaciones sobre el que trabajará el protocolo 11073.

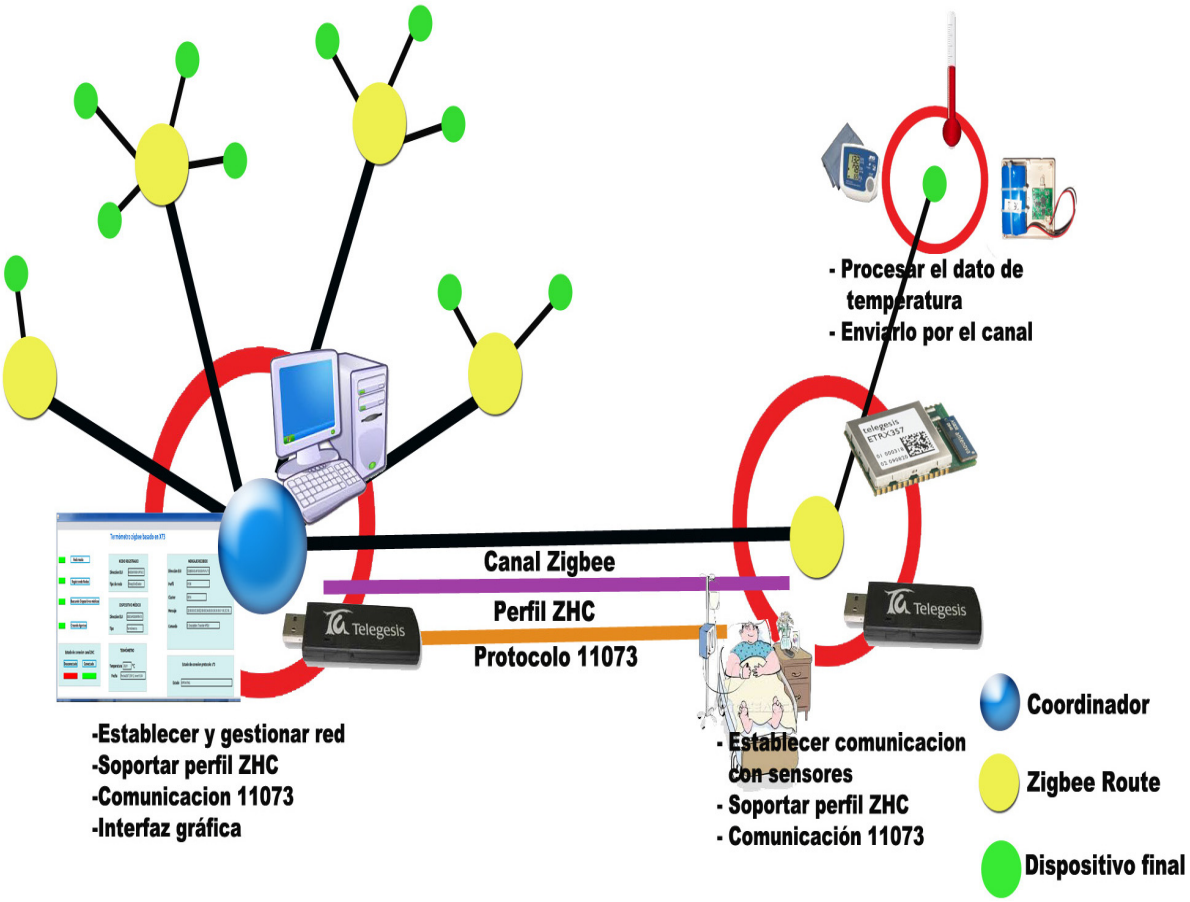


Figura 1.1 Resumen gráfico del proyecto

Finalmente, una vez establecido el canal se pondrá en funcionamiento una máquina de estados, en ambos extremos, encargada de implementar todos los pasos correspondientes al protocolo 11073, para finalmente transmitir el dato de temperatura de un modo seguro.

A parte de todo lo anterior parar poder desarrollar el proceso de un modo correcto, también será necesario enviar los datos de acuerdo al tipo de trama correspondiente a cada nivel, ya que la comunicación a nivel de 11073 se lleva a cabo mediante tramas APDU, y para tratar con un perfil público de ZigBee Alliance es necesario gestionarlo a través de las tramas correspondientes al ZCL (ZigBee Cluster Librery) [8].

De este modo una vez finalizado el proyecto quedará definida toda la lógica necesaria para que se pueda implementar un agente y un manager que trabajen por sí solos dentro de una entorno médico y estaremos en disposición de trasladar todo el software de gestión del agente al propio sensor.

Todo lo expuesto anteriormente se gestionará por medio de una serie de bloques de programación, encargados de gestionar las funciones correspondientes tanto al manager como al agente, realizando todo el sistema de un modo automático, a través de un entorno Java. De un modo paralelo se desarrollará una interfaz gráfica que se implementará en el extremo correspondiente al coordinador y mostrará por pantalla todas las fases del proyecto, creación de la red, creación del canal ZHC y los estados del protocolo X73, para finalmente mostrar el dato de temperatura proveniente del sensor.

Por último, ya que todo el PFC se desarrolla en un entorno de código abierto, se tratará de que una vez finalizado el sistema quede completamente público para que se pueda integrar con otras implementaciones de software libre.

1.5 Estructura de la memoria

Esta memoria ha sido estructurada en ocho apartados y cuatro anexos, como se explica a continuación:

- **1. Introducción**: Se trata del apartado en el que nos encontramos y busca presentar el proyecto, el lugar de donde se parte, así como descubrir los objetivos que se pretender alcanzar.

- **2. Estado del arte**: En este apartado se realiza un estudio de cuál es la situación actual de todas las tecnologías empleadas en el proyecto, el análisis se lleva a cabo tanto a nivel tecnológico, como social o legislativo presentando los organismos que regulan la estandarización. También se muestran el tipo de dispositivos que podemos encontrar actualmente en el mercado.

- **3. Materiales y métodos**: Este apartado será el encargado de mostrar una visión técnica de todas las tecnologías que se van a emplear, así como los protocolos que se van a implementar y finalmente una descripción de los dispositivos utilizados a lo largo del PFC.

- **4. Análisis y diseño**: Aquí es donde se va presentar la solución aportada, la arquitectura utilizada para llevar a cabo el proyecto, centrándonos en los tres grandes bloques que lo componen que son la tecnología ZigBee, el perfil ZHC y el protocolo 11073.

- **5. Desarrollo e implementación:** En este capítulo se va presentar todo el entorno de trabajo donde se ha llevado a cabo el proyecto y se van a presentar las partes principales que componen el software que se ha desarrollado.

- **6. Resultados:** En este apartado se muestran los resultados obtenidos una vez finalizado el proyecto.

- **7. Conclusiones y líneas futuras:** En este apartado realizamos un análisis de las conclusiones obtenidas una vez finalizado el PFC, asi como las posibles líneas de trabajo que nos quedan abiertas una vez acabado.

- **8. Bibliografía:** Se recoge toda la documentación utilizada para llevar a cabo el proyecto**.**

- **A1. Anexo 1- Cronograma de implantación:** En este apartado se presentan los tiempos que se han empleado para desarrollar cada parte del proyecto.

- **A2. Anexo Configuración de registros:** Se recoge en dos tablas los valores que toman cada uno de los registros que configuran los módulos de Telegesis.

- **A3. Anexo 2 Interfaz Gráfica:** Se explica el funcionamiento de la interfaz gráfica diseñada cuyo objetivo es mostrar el funcionamiento del manager y la medida del dato de temperatura.

- **A4. Anexo 4 Emulado del agente:** Se expone de un modo breve el proceso que se ha seguido para conseguir combinar uno de los módulos de Telegesis y el sensor de temperatura para que funcionen como un agente.

2 Estado del arte

En este apartado nos vamos a centrar en realizar un estudio del panorama actual en lo referente a la telemedicina y la tecnología ZigBee así como un análisis de la situación actual de las tecnologías empleadas, además de todos los avances desarrollados en los últimos años que atañen a este proyecto.

2.1 Telemedicina y estandarización

Uno de los grandes retos de este siglo es el poner al alcance de todo el mundo un sistema de control de la salud de calidad. Una tarea que puede resultar difícil o quizás imposible si tenemos en cuenta el incremento en la población mundial, el aumento del envejecimiento o la aparición de nuevas enfermedades. No obstante también hay que tener en cuenta los avances de la tecnología ya que aquí se encuentra la solución para dicho reto y esto es algo que se han dado cuenta los gobiernos, lo que se ha traducido en una fuerte inversión en el sector de la telemedicina a nivel internacional con el objetivo de modernizar el sistema.[9]

En un artículo publicado en El País en agosto de 2012 basado en un informe de la BBC Research [10], se afirmaba que la telemedicina y el negocio de la e-salud iba a triplicar sus servicios en un plazo de seis años. Otro informe reciente elaborado por Pike & Fischer [11], estima que los gastos en telemedicina alcanzarán los 3.600 millones de dólares estadounidenses anualmente para el año 2014. Además de esto, en el año 2009 en EEUU se creó un plan de recuperación y reinversión que incluyó 20.000 millones de dólares estadounidenses para la tecnología de información sanitaria, con un enfoque en los registros médicos electrónicos y la telemedicina, el cual viene recogido en el de American Recovery and Reinvestment Act de 2009 [12]. Por último, otro ejemplo de la inversión que se está llevando a cabo en los últimos años en la telemedicina lo podemos encontrar en el Reino Unido, donde el gobierno se ha centrado en incluir la telemedicina como una estrategia del gobierno para mejorar el servicio nacional de salud (NHS).

Por otro lado, tenemos que todos los desarrollos de aplicaciones vienen marcados por la estandarización del producto, con el fin de buscar un entorno normalizado de trabajo y ofreciendo una opción para que se puedan crear diferentes líneas de desarrollo independientes a los fabricantes. Así pues antes de meternos a analizar el estándar que vamos a utilizar, puede resultar de utilidad conocer que organismos son los encargados de regular estas normas, los podemos encontrar tanto a nivel nacional como internacional. En el caso de los dispositivos biomédicos, los organismos que regulan estas normas son AENOR (Asociación Español de Normalización y Certificación) en el ámbito nacional y el CEN (Comité Europeo de Normalización) a nivel internacional, desde la década de los 90. Otro de los organismos de regulación que nació tras la Segunda Guerra Mundial, con el nombre de ISO (Organización Internacional de Normalización) y es el la encargada de promover el desarrollo de normas internacionales de fabricación [13].

En cuanto a los estándares que rigen el intercambio de información en procesos de telemedicina destacan HL7 [14], que se trata de una organización que ha lanzado una serie de estándares para facilitar el intercambio de información en salud, los cuales han sido aprobados por la ANSI (American

National Standar Institute).

Por lo tanto debido al alto número de estándares y con el fin de que se cumplan todos ellos y exista una interoperabilidad total entre dispositivos, nace Continua Health Alliance [15] que se trata de una asociación sin ánimo de lucro formada por un conjunto de más de 230 empresas, cuya misión es establecer un ecosistema de interoperabilidad entre dispositivos médicos.

2.2 ZigBee y ZigBee Cluster Library

ZigBee se trata de una tecnología relativamente nueva promovida por ZigBee Alliance [6], un conjunto de más de 100 empresas sin ánimo de lucro entre las que destacan empresas como Invensys, Mitsubishi, Honeywell, Philips o Motorola entre otras [16]. Si hacemos un pequeño repaso por la historia de la tecnología ZigBee vemos como comenzó a desarrollarse en 1999 cuando se dieron cuenta de que las tecnologías Bluetooth y Wi-fi no resultaban prácticas en determinados escenarios. Más tarde en mayo de 2003 se publicó el estándar IEEE 802.15.4 [17] un estándar que define el nivel físico y el control de acceso al medio de redes inalámbricas con tasas bajas de transmisión de datos. No obstante, no fue hasta junio de 2005 cuando se puso la tecnología ZigBee al alcance del público. Posteriormente en 2007 se hizo pública la aplicación ZigBee Home Automation de un modo gratuito y posteriormente se han ido publicando nuevos perfiles compuestos por conjuntos de bloques comunes que vienen recogidos en la ZigBee Cluster Library [8] , actualmente los que se encuentran disponibles son:

- ZigBee Home Automation
- ZigBee Smart Energy 1.0
- ZigBee Telecommunication Services
- ZigBee Health Care
- ZigBee RF4CE – Remote Control
- ZigBee RF4CE – Input Device
- ZigBee Light Link

También existen algunas especificaciones en desarrollo como son:

- ZigBee Smart Energy 2.0
- ZigBee Building Automation

A día de hoy se prevé, que la tecnología ZigBee sea adoptada en todo el mundo como la principal tecnología inalámbrica estándar de sensores y controles en la energía de consumo, especialmente en el caso de la domótica y la telemedicina. Esto viene apoyado por importantes acuerdos como el que existe en la actualidad entre ZigBee Alliance y ATA (American Telemedicine Asociation) que trata la formación tanto de profesionales como de los propios consumidores [18].

Ya que este PFC está pensado para un ámbito médico vamos a implementar el perfil para controlar dispositivos médicos que nos proporciona ZigBee Alliance, este perfil es el ZigBee Health Care [7], el cual, en Junio de 2009, recibió el respaldo por parte de Continua Health Alliance [15] como el estándar para redes de área local de bajo consumo en el estándar Continua 2010 Design Guidelines [19], aunque no fue hasta abril de 2010 cuando se ZigBee Alliance lo público como estándar global que permite la monitorización y gestión segura de datos no críticos.

Así pues podemos concluir que si la tecnología ZigBee es nueva, el perfil médico que la rige todavía más y esto hace que se trate de un campo donde todavía queda mucho por desarrollar, no obstante en la actualidad podemos encontrar algunos proyectos que ya han desarrollado dispositivos con el ZHC integrado, como los desarrollados por la universidad de Brunel (Reino Unido) que podemos observar en la figura 2.1. y que podemos consultar a través de la propia página de ZigBee Alliance [6].

Figura 2.1- Dispositivos que soportan el ZHC

Además si realizamos una rápida búsqueda por la página de Continua [15] con el objetivo de encontrar los dispositivos que han sido certificados en los últimos años, obtenemos que solo se han registrado ocho en los últimos dos años, la mayoría por parte de la ya mencionada universidad de Brunel, y uno por InfoSolutions, que se trata de una compañía italiana. Lo que nos dice que es un mercado que se encuentra en vías de desarrollo y todavía por explotar.

Search Results 8 Results

Device Type	Manufacturer	Product Name	Cert. Date
Pulse Oximeter	Info Solution SpA	OxSys Pulse Oximeter	2012-06-25
Glucose Meter	Brunel University	Ultra 2 Blood Glucose Reader	2012-04-28
Pulse Oximeter	Brunel University	Nonin Ipod Pulse Oximeter	2012-04-28
Independent Living Activity Hub	Brunel University	ZTX450Z Sensor Controller	2012-04-27
Blood Pressure Monitor	Brunel University	A&D Blood Pressure Monitor	2012-02-01
Independent Living Activity Hub	Brunel University	Carousel Mk 3 Pill Dispenser	2012-02-01
Independent Living Activity Hub	Brunel University	EX-35R Motion Sensor (PIR)	2012-02-01
Weighing Scale	Brunel University	A&D UC-321PZ-C Weighing Scale	2012-01-24

Figura 2.2 – Dispositivos Certificados por Continua Alliance

2.3 Protocolo X73

Uno de los graves problemas de los dispositivos médicos es la variedad de protocolos de comunicaciones establecidos por los proveedores. Este es el motivo por el que surge la familia de estándares *X73 Medical / Health Device Communication Standards*, la cual pretende estandarizar las comunicaciones de dispositivos médicos, abstrayéndolos del canal de comunicaciones establecido y permitiendo la interoperabilidad de los mismos, ofreciendo de este modo una solución a dichos problemas de interoperabilidad.

El proyecto se inició en el año 2000 como un nuevo proyecto común en un intento por crear un solo estándar, esta vez de categoría ISO y el trabajo culminó en 2004 con la publicación de la primera serie de la familia CEN/*ISO/IEEE 11073 Health Informatics – Point-of-Care Medical Device Communication* (X73PoC). Más adelante en 2008, una nueva versión de X73 para un contexto de aplicación de salud más personalizado es publicada bajo la denominación de *ISO/IEEE 11073 Health Informatics – Personal Health Device Communication (X73PHD)***,** siendo esta la versión más reciente.

Un esquema de pilas de protocolos de ambas versiones y su arquitectura de modelo/servicio se muestra en la Figura 2.2 [20].

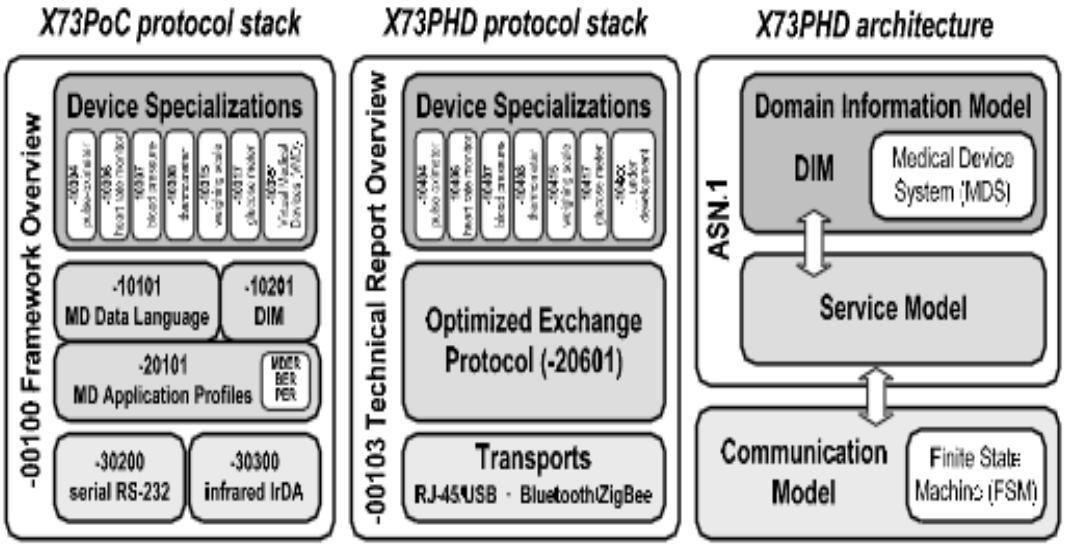


Figura 2.3. Evolución de la pila de protocolos de X73PoC a X73PHD.

En la actualidad podemos encontrar diferentes tipos de dispositivos médicos que han recibido la certificación por Continua Health Care y trabajan bajo el estándar 11073. En la figura 2.3 se muestran dos ejemplos de dispositivos médicos que además de realizar la medida del dato clínico, establecen una comunicación con el usuario a través de tecnología Bluetooth y basando su comunicación en la norma X73, ambos dispositivos han recibido la certificación por parte de Continua Health Alliance.

Figura 2.4 Dispositivos médicos con la normas X73 certificados por Continua Health Alliance

De un mismo modo que en el caso de la tecnología ZigBee si realizamos una búsqueda en la página de Continua, vemos existe un amplio catálogo de dispositivos registrados que soportan el protocolo 11073, aunque la mayoría de ellos a excepción de los mostrados en la figura 2.2 no trabajan con la tecnología ZigBee.

Search Results 3 Results

Device Type	Manufacturer	Product Name	Cert. Date
Glucose Meter	MindTree	EtherMind IEEE-11073 Glucosemeter Agent	2011-06-03
Weighing Scale Pulse Oximeter Blood Pressure Monitor Thermometer Cardiovascular Strength Independent Living Activity Hub Adherence Monitor Peak Flow Monitor Glucose Meter	MindTree	EtherMind IEEE-11073 Manager	2011-05-31
Weighing Scale Pulse Oximeter Blood Pressure Monitor Thermometer Cardiovascular Strength Independent Living Activity Hub Glucose Meter	Intel	Intel® Evaluation Kit with IEEE 11073 Continua Certified Software Stack for Medical Applications	2011-04-21

Figura 2.5 Productos que soportan 11073 certificados por Continua

3 Materiales y métodos

En este apartado se muestran unas orientaciones teóricas respecto a las diferentes tecnologías que se han empleado, así como los diferentes materiales manejados, con el fin poder comprender de una mejor manera el análisis del proyecto.

3.1 ZigBee

Se trata del pilar fundamental del proyecto, el hecho de haber escogido este tipo de comunicación inalámbrica y no otra se debe a que resulta idónea para el envío de datos provenientes de sensores, es más, ZigBee fue específicamente diseñada pensando en la proliferación de sensores individuales que se estaban comercializando.

La finalidad de la comunicación ZigBee es presentar una tecnología de bajo coste, siendo un estándar para redes de pequeños paquetes de información, seguro, fiable y de bajo consumo que se traduce en la maximización de la vida útil de las baterías.

En los siguientes puntos vamos a presentar las características principales que engloban esta tecnología.

3.1.1 Arquitectura de ZigBee

ZigBee presenta una arquitectura basada en el modelo de referencia OSI compuesta por diferentes capas como se observa en la figura 3.1.

Las dos capas más bajas de la comunicación son, la capa física y la capa de control de acceso al medio (MAC). La capa física (PHY) trabaja a unas frecuencias de 2.4 GHz en todo el mundo, 868 MHz en Europa y 915 MHz en Estados Unido, está compuesta por hasta 16 canales, con un ancho de banda de 5MHz cada uno y junto con la capa de control de acceso al medio nos proporcionan los servicios de la comunicación inalámbrica. El rango puede variar de 10 a 100 metros en función de las potencias utilizadas y el entorno en el que nos encontremos.

Mientras que la capa de control de acceso al medio nos proporciona los servicios necesarios para permitir una comunicación directa entre dispositivos, la capa de red resulta necesaria para ofrecer servicios a la capa inmediatamente superior, la capa de Aplicación, y que su vez permitan realizar operaciones sobre la capa inmediatamente inferior a la misma, la capa de MAC. Es decir, la capa de red hace de interfaz entre la capa de Aplicación y la de MAC.

También existe otra capa antes de llegar a la capa de aplicación que se denomina GOF y es la encargada de cubrir las tareas correspondientes a los modos de direccionamiento.

Por último en la capa de aplicación encontramos el punto de interconexión entre el nodo ZigBee y los usuarios y será sobre esta capa donde centremos el trabajo de este PFC [21].

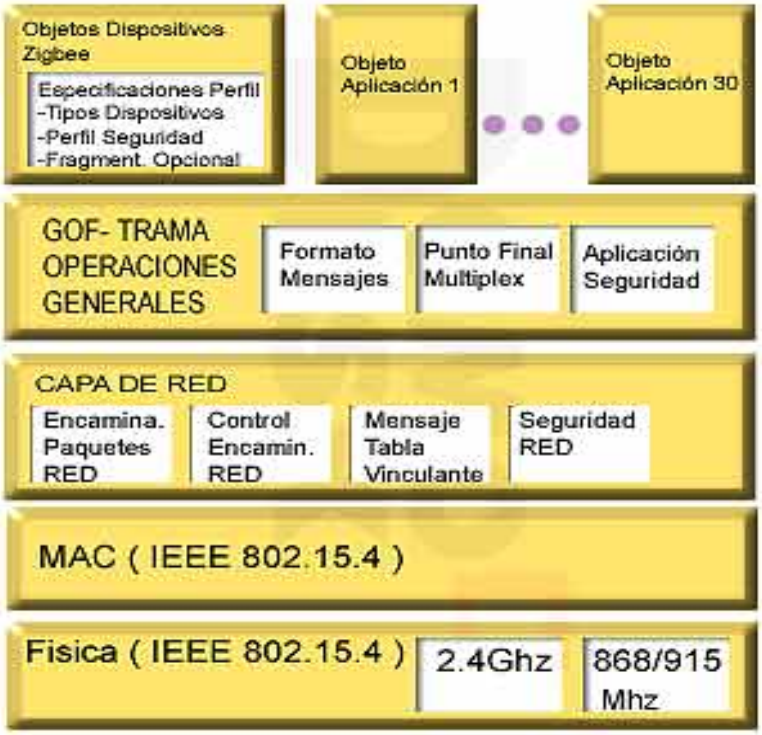


Figura 3.1 Capas arquitectura ZigBee

3.1.2 Tipos de dispositivos

Otra característica importante es definir los distintos tipos de dispositivos que podemos encontrar dentro de una red ZigBee, según su función dentro de la red, son:

- *Coordinador*: Se encarga de controlar la red y caminos a seguir por los dispositivos para conectarse entre ellos
- *Router*: Interconecta los dispositivos separados en la red y ofrece un nivel de aplicación para el usuario
- *EndDevice*: Se comunica con su nodo padre (COO o FFD) por ejemplo en nuestro caso el sensor de temperatura.

Toda red debe de estar compuesta por un Coordinador y tantos routers como endpoints nos permita la red, de este modo podemos formar una red con distintas topologías:

- Topología en estrella: el coordinador se sitúa en el centro.
- Topología en árbol: el coordinador será la raíz del árbol.
- Topología de malla: al menos uno de los nodos tendrá más de dos conexiones.

En la figura 3.2 se muestra como quedarían los diferentes dispositivos dentro de las posibles redes ZigBee que nos podemos encontrar.

En este PFC trabajaremos con una red compuesta por un coordinador, un router y un dispositivo final.

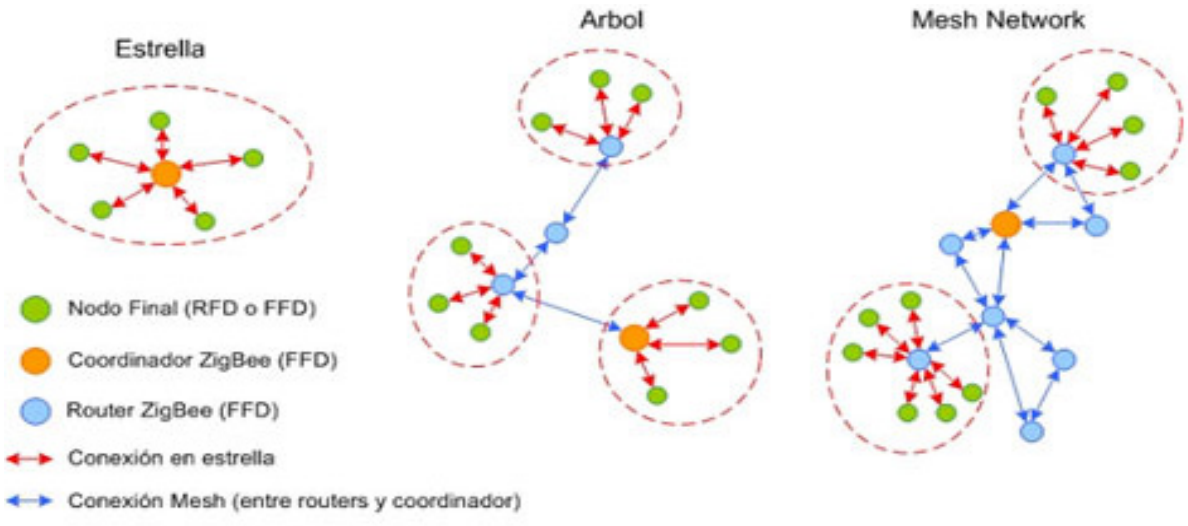


Figura 3.2 Dispositivos dentro de una red ZigBee

3.1.3 Endpoints y clusters

La comunicación en los dispositivos ZigBee se produce entre *endpoints* a través de unas estructuras de datos que se llaman *clusters*, los cuales están compuestos por una serie de atributos correspondientes a cada perfil.

Ya que un bloque fundamental en el proyecto es la configuración tanto de clusters como de endpoints, resulta necesario realizar una introducción de ambos.

3.1.3.1 Endpoints

Cada nodo ZigBee está compuesto por un número determinado de endpoints que van desde el 0 utilizado para configuraciones y mantenimiento hasta el 255, a través de estos endpoint podemos establecer la comunicación con los endpoint de otro dispositivo, cada uno de ellos define una aplicación diferente dentro del nodo, permitiéndonos así la posibilidad de que un nodo pueda contener varios perfiles.

Por ejemplo en este PFC en el endpoint 1 del dispositivo ZigBee se ubica el perfil correspondiente a la gestión de Telegesis mientras que utilizaremos el endpoint 2 para colocar el perfil ZHC. Como ya se ha indicado de este modo podemos establecer una comunicación entre endpoints de diferentes dispositivos, en nuestro caso lo haremos entre los endpoint que contengan el perfil médico ZHC.

Por su parte también cabe destacar que cada dispositivo tiene una aplicación especial en el endpoint 0 denominada ZDO (ZigBee Device Object), que es la que se encarga de las tareas de configuración y funcionamiento automático, constituye una especie de comunicación transversal donde puede controlar parámetros de la capa de red (NWK) y soporte de aplicaciones (APS) el perfil de aplicación que contiene es el ZDP (ZigBee Device Profile) [22].

3.1.3.2 Clusters

En cambio los clusters los podemos definir como aplicaciones dentro de un perfil determinado, se definen por identificadores de 16 bits y cada nodo está compuesto por uno o varios clusters que a su vez contienen una serie de atributos y comandos propios de cada perfil para poder gestionarlos.

Un ejemplo de esto lo podemos encontrar en la tabla 3.1 donde se muestran algunos de los diferentes clusters que se utilizan para la gestión de los perfiles públicos que nos proporciona ZigBee Alliance, los cuales se encuentran publicados en la ZigBee Cluster Library [8].

Cluster ID	Cluster Name	Description
0x000	Basic	Nos proporciona los atributos basicos respecto a la información del dispositivo
0x001	Power Configuration	Contiene los atributos que determinan información mas detallada sobre la batería del dispositivo y configuraciones a cerca del voltaje
0x002	Device Temperature Configuration	Los atributos que determinan la información sobre la temperatura interna del dispositivo
0x003	Identify	Dispone de los atributos necesarios para poner un modo de identificación en el dispositivo(e.g. encender una luz)

Tabla 3.1 Ejemplo de Clusters

3.1.4 Características y comparativa

Aquí tratamos de dar respuesta al porque esta tecnología resulta idónea para este tipo de comunicación comparándola con otras tecnologías inalámbricas como son Bluetooth o Wi-fi.

En primer lugar ZigBee nos presenta la posibilidad de que coexistan varias redes ZigBee en un mismo espacio, ya que cada red posee un único identificador de bits llamado PAN ID.

Por otro lado en lo que respecta a los nodos que componen la red, en una red ZigBee podemos conectar hasta 254 nodos con los 250 endpoints y dentro de cada nodo podemos llegar a tener hasta 64770 frente a los 8 nodos por subred que encontramos en una subred Bluetooth.

En cuanto a la tasa de envío de datos como se puede observar en la tabla 3.2 vemos como se trata de una tasa mucho menor respecto a Wi-fi o Bluetooth, pero resulta suficiente para realizar este tipo de tareas de intercambio de información con sensores, además el hecho de que se trate de una tasa de envío tan baja le permite la coexistencia con las tecnologías WiFi y Bluetooth a pesar de trabajar en la misma frecuencia.

	ZigBee	Bluetooth	Wi-fi
Tasa de envío de datos	250 Kb/s	1Mb/s	54Mb/s

Tabla 3.2 Tasas de envío de datos ZigBee, Bluetooth y WiFi

Otra de las ventajas que nos ofrece una red ZigBee es que gracias a su

tecnología de salto puede llegar a alcanzar nodos que estén fuera de su rango, así como solucionar caídas o fallos del sistema gracias la existencia de múltiples rutas.

Y por último, una de las cualidades más importantes de ZigBee es la duración de sus baterías, ya que tiene la característica de que los nodos finales de una red permanezcan prácticamente todo el tiempo dormidos y únicamente despierten durante una fracción de segundo cuando tiene que establecer la comunicación, además la capacidad de memoria que necesita es muy pequeña lo que hace disminuir los costes en gran medida.

3.1.5 ZigBee Cluster Library (ZCL)

Para entender mejor el funcionamiento de los perfiles de ZigBee es necesario hacer una pequeña introducción de la ZigBee Cluster Library [8], se trata de una librería que nos proporciona ZigBee Alliance y se compone de una serie de estándares ya definidos a partir de los cuales podemos montar nuestras propias aplicaciones.

Por otro lado también nos describe una serie de perfiles con los que podemos trabajar, que a su vez se apoyan en los clusters de la ZCL y fijan tanto los comandos como la funcionalidad del dispositivo en un entorno determinado, actualmente se disponen de los siguientes perfiles:

- Smart Energy (SE)
- Home Automation (HA)
- Commercial Building Automation (CBA)
- Telecom Applications (TA)
- Personal, Home and Hospital Care (ZHC)

3.1.6 ZigBee Health Care (ZHC)

En nuestro caso vamos a trabajar con el perfil ZigBee Health Care [7], que se trata de un estándar que nos ofrece ZigBee Alliance y ha sido diseñado específicamente para trabajar en un entorno médico como residencias de ancianos, centros deportivos o monitorización de pacientes.

Se trata de un perfil que aparte de proporcionarnos una serie de comandos, atributos y descripciones de dispositivos estándares para trabajar con ZigBee, también sirve como soporte para el estándar IEEE 11073.

A su vez este perfil nos indica entre otras especificaciones el descriptor que debemos usar para diferenciar los diferentes tipos de dispositivo (tabla 3.3), en nuestro caso se tratará del termómetro. Los clusters obligatorios que tienen que soportar en ambos extremos del canal (tabla 3.4) , así como todas las instrucciones que coordinan cada uno de estos clusters y también indican los estados qu

e deben seguir cada dispositivo y otras muchas instrucciones y recomendaciones para finalmente acabar siendo el soporte de la norma X73.

	Device Description	Device ID
Disease Management	Pulse Oximeter	0x1004
	ECG	0x1006
	Blood Pressure Monitor	0x1007
	Thermometer	0x1008
	Weight Scale	0x100f
	Glucose Meter	0x1011
	International normalized ratio (INR)	0x1012
	Insulin Pump	0x1013
	Peak Flow Monitor	0x1015

Tabla 3.3- ID de los dispositivos

Server side	Client side
Mandatory	
Basic	-
Identify	-
Generic Tunnel	
11073 Protocol Tunnel	11073 Protocol Tunnel

Tabla 3.4- Clusters soportados en ambos lados del ZHC

3.2 Norma ISO/IEEE 11073

Con la finalidad de dotar al proyecto de interoperabilidad entre el agente y el manager se ha optado por implementar la norma ISO/IEEE 11073, sobre el perfil ZHC. Dicha norma se trata de un conjunto único de especificaciones desarrolladas con el objetivo de garantizar la conectividad completa entre dispositivos médicos, aportando capacidades de interoperabilidad, transparencia y facilidad de uso, también se conoce este estándar como la norma X73.

Esta norma surgió a raíz de una iniciativa de unificación de una serie de normas ISO e IEEE previas con el objetivo de integrar en una sola especificación todos los niveles en la comunicación entre los dispositivos.

Se gestiona a través de una máquina de estados finitos que define todos los estados por los que pasa la comunicación que son: desconectado, conectado, desasociado, asociado, asociando configurando y operando. En la figura 3.3 se muestra un diagrama de cómo gestionan estos estados durante el proceso de comunicación entre un agente y un manager.

La norma 11073 funciona del siguiente modo, en primer lugar una vez establecido el canal ZHC ambos extremos parten en estado desconectado, el agente tratará de establecer la conexión y en el caso de que se concluya con éxito pasarán al estado conectado pero desasociados, para asociarse deberán de pedir una conexión de asociación, si el manager conoce la configuración ambos dispositivos quedarán asociados , en caso contrario le pedirá la configuración y guardará los parámetros necesarios pasando así al estado operando, donde se realizará la codificación y el intercambio de los datos de

temperatura. Una vez alcanzado el estado operando en cualquier momento se puede establecer una petición de desasociación, bien de modo voluntario o involuntario [23-24].

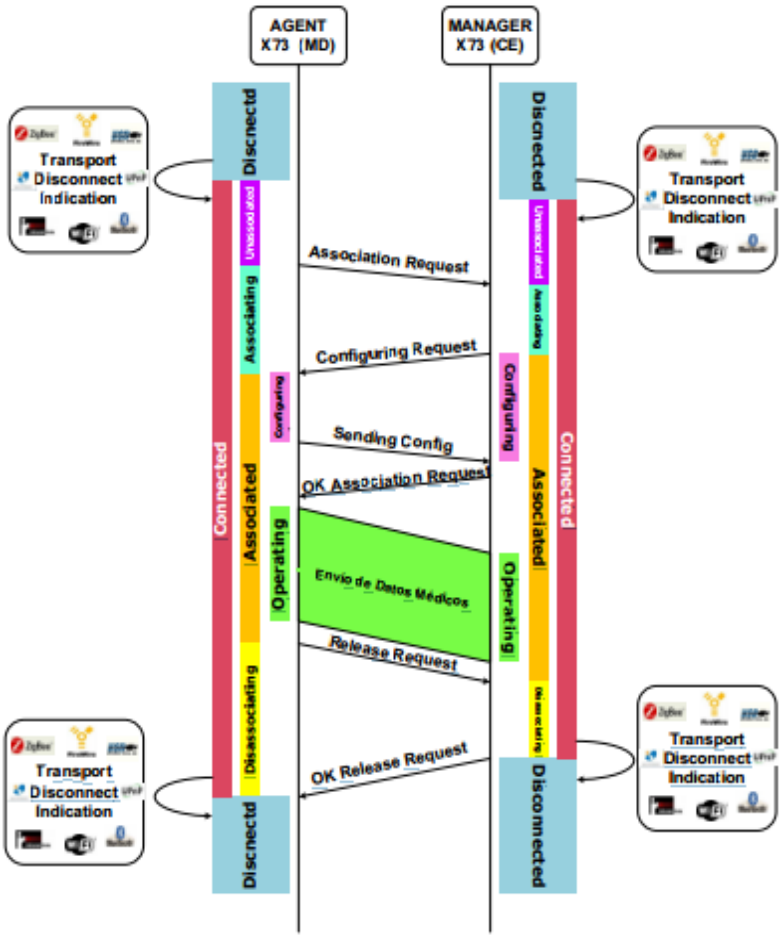


Figura 3.3 Esquema de la comunicación MD-CE

Para gestionar todo el intercambio de información que se produce entre el manager y el agente, se lleva a cabo a través tramas APDU (Application Protocol Data Unit) que a su vez pueden ser de diferentes tipos, según el cometido de dicha trama, estos tipos son:

- AARQ: Solicitud de asociación
- AARE: Respuesta a la solicitud de asociación
- RLRQ: Liberar la asociación
- RLRE: Respuesta a la petición de liberación
- ABRT: Abortar comunicación
- PRST: Dato APDU

3.3 Material utilizado

Una vez presentadas las tecnologías que se van a emplear y el medio de comunicación a través del cual se va a establecer la comunicación, en este punto se presentan los dispositivos que han sido empleados para llevar a cabo el PFC.

3.3.1 Modulo ETRX2

Para realizar la función del coordinador y router se han utilizado dos módulos de la empresa Telegesis, más concretamente el producto ETRX2USB que podemos ver en la figura 3.5 (en [25] y [26] se encuentran las especificaciones del producto y el manual de utilización respectivamente mientras que los drivers del dispositivo se pueden descargar de la página de Telegesis: www.telegesis.com).

El ETRX2 trabaja a 2.4 GHz, por lo tanto cumple con el estándar 802.15.4 y tiene la posibilidad de configurarlos como Coordinador, Router o End Device, también nos permite el acceso a conjunto de registros para gestionar su configuración.

Han sido necesarios los dos dispositivos para llevar a cabo el proyecto ya que uno lo configuraremos como el coordinador de la red mientras que el otro dispositivo además de actuar como un router será el encargado de junto con el sensor de temperatura emular el comportamiento de un agente médico, como se explica en el anexo 4.

En la tabla 3.5 se muestra un resumen de las principales características del dispositivo de Telegesis.

Figura 3.6 Modulo ETRX2

P salida máxima	250 Kbps
Alcance	> 150 m
Tasa de datos	250 Kbps
Tensión de alimentación	2,1 - 3,6V
Frecuencia	2,4GHz
Temperatura	-40ºC +85ºC
Dimensiones	L82mm x W26mm x H8.5mm
Coste	27 €

Tabla 3.5- Características del módulo ETRX2

Cada uno de los dispositivos dispone de 79 registros que van desde del S00 al S4F, los cuales pueden ser configurados en función de las funciones que se quiere que realicen.

El comportamiento de los dispositivos se controla a través de los comandos AT [27], como por ejemplo en el caso de la formación de la red (AT+EN) o unirse a la red (AT+JN). A parte en el caso de enviar los mensajes podemos hacerlo de dos maneras, por un lado mensajes broadcast, mediante el comando AT+BCAST donde envía la información a todos los dispositivos de la red, y por otro lado mediante el comando AT+UCAST: <dirección>, que envía la información a un dispositivo concreto.

3.3.2 Sensor de temperatura

Como dispositivo final se va a utilizar un módulo facilitado por el grupo de investigación Howlab (Figura 3.6).

El hardware que compone el sensor está compuesto por compuesto por un microcontrolador PIC18F26J11, un sensor de temperatura y humedad, así como un módulo ZigBee.

Será el encargado de tomar la medida de la temperatura y transmitirla por medio del módulo ZigBee hasta uno de los módulos ETRX2 (Anexo 4).

Una información más detallada acerca del sensor se puede consultar en la página de openlab.unizar bajo el nombre del proyecto ZetaMotaAmbi. [28].

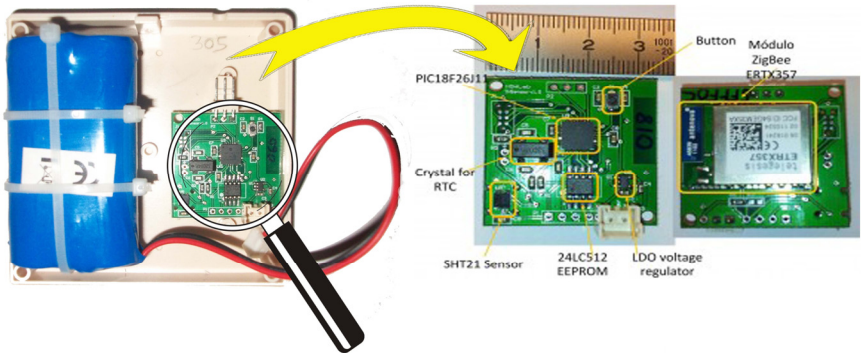


Figura3.7 Sensor ZetaMotaAmbi

4 Análisis y diseño

Una vez explicada toda la tecnología y el material utilizado, en este apartado pasamos a dar una visión más global del sistema, y explicar de un modo detallado todos los pasos que se han seguido para llevar a cabo el proyecto.

4.1 Análisis

Lo que se pretende conseguir con este PFC es el establecer una plataforma de trabajo, donde se pueda conectar un dispositivo médico, en nuestro caso un sensor de temperatura, que actuará como un agente, con un dispositivo central que cumpla la funcionalidad de manager. Todo esto a través de una red de comunicación inalámbrica ZigBee.

Para poder desarrollar el proyecto de un modo correcto, ya que no disponemos de un dispositivo médico homologado se ha optado por la opción de emular dicho agente mediante la combinación de uno de los dispositivos ETRX2 [25] de Telegesis y el sensor de temperatura [28] como se muestra en la figura 4.1.

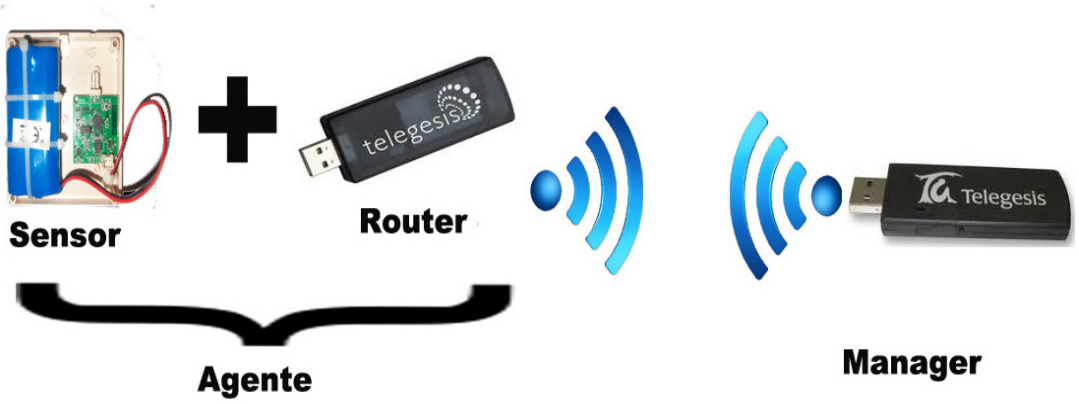


Figura 4.1- Esquema del montaje

De este modo obtendremos por un lado un dispositivo capaz de crear y gestionar una red ZigBee, y que a su vez que pueda soportar el protocolo X73. Mientras que por otro lado obtenemos la arquitectura necesaria para poder implementar un agente médico, pensando que en una segunda versión del proyecto, una vez obtenido el modo de trabajo y todos los ajustes necesarios, se traslade la configuración del router al propio sensor para que éste pueda trabajar de un modo individual como un agente médico.

Así pues el diseño que se ha decidido seguir se presenta en la figura 4.2, donde se muestra cuáles son los niveles que implementará cada dispositivo, vemos como el sensor solo se encarga de tomar y enviar la temperatura al router y éste es el encargado de establecer el canal de comunicación con el coordinador y sobre este transmitirle el dato de temperatura, mientras que el coordinador además de crear la red y establecer la comunicación con el router, va a disponer de una interfaz gráfica que nos va a mostrar desde el proceso de asociación hasta el dato de temperatura (Anexo 3).

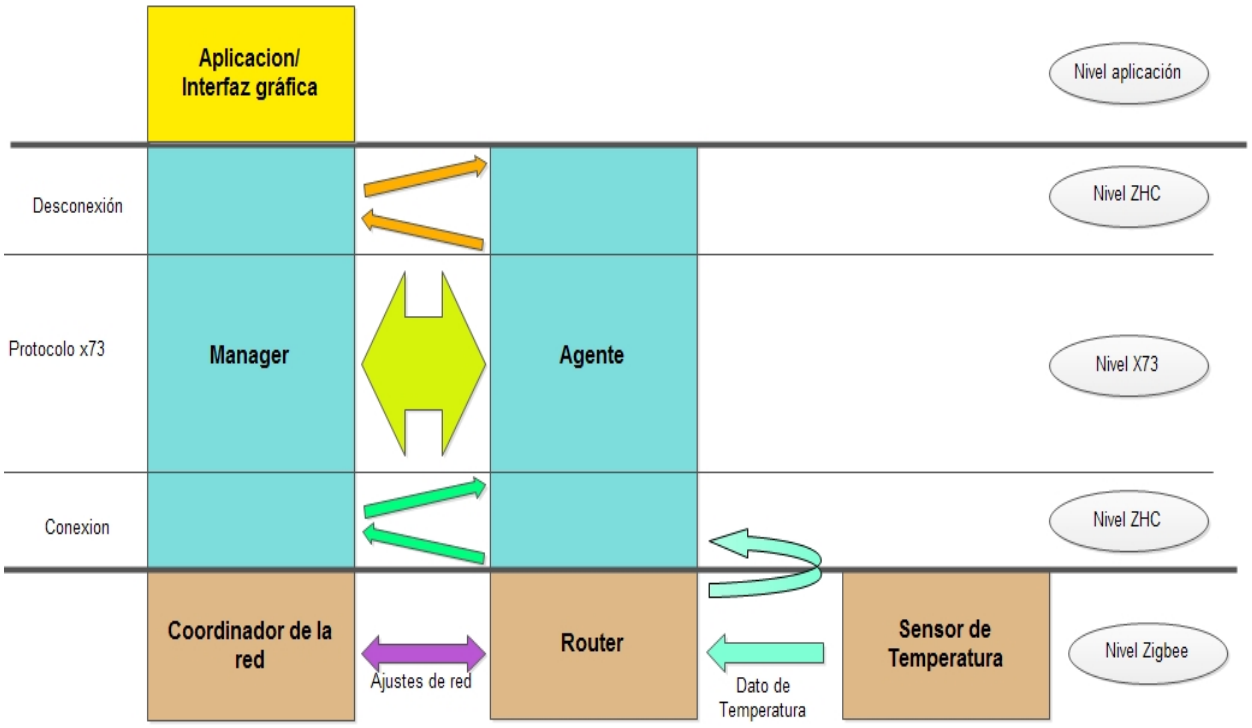


Figura 4.2 Arquitectura del modelo

Una vez que ha quedado clara la función de cada uno de los dispositivos dentro de la red pasamos a analizar el trabajo que se ha llevado a cabo en cada una de las capas.

En primer lugar en la capa correspondiente a ZigBee tenemos que estudiar una configuración de los dispositivos ETRX2 para que cada uno actúe de acuerdo a la función que le corresponde, luego pasaremos a desarrollar un software que generé el proceso de creación y gestión de la red de un modo automático, para lo que nos apoyaremos en el Zeta Project [29], un proyecto de código abierto que nos proporciona las herramientas necesarias para mandar las instrucciones a los módulos ZigBee.

Por otro lado para implementar el perfil ZHC, también será necesaria la configuración de determinados registros, así como desarrollar sobre la capa anterior un sistema que sea capaz de implementar a ambos lados de la comunicación los clusters que componen el perfil ZHC.

Por último, el sistema será capaz de desarrollar sobre el canal ZHC, todo el protocolo de comunicación correspondiente a la norma X73 gestionando una máquina de estados finita que implemente todas las fases de la comunicación.

En los siguientes apartados se explica de un modo más detallado cada uno de los bloques que componen el proyecto.

4.1.1 Arquitectura ZigBee

Para llevar a cabo el desarrollo de esta primera fase del proyecto, fue necesario en primer lugar realizar diferentes pruebas para comprender el funcionamiento tanto de los dispositivos en particular como de la tecnología ZigBee en general. Una vez comprendido el funcionamiento de la tecnología ZigBee se pasó a implementar la automatización del sistema, además como ya se ha explicado en el apartado de materiales y métodos, fue necesaria realizar una configuración de cada uno de los registros que posee el dispositivo y configurarlos para que soporte el perfil ZHC.

4.1.1.1 Configuración de registros

Para que la comunicación se pueda dar de acuerdo a la arquitectura expuesta, ha habido que realizar un estudio de cada uno de los registros S para configurarlos de acuerdo al cometido que van a tomar en la red y poder aplicarle estos cambios cuando sea preciso, por lo tanto existe una configuración diferente para cada uno de los dispositivos.

En este apartado se detallan los cambios más significativos, no obstante, para una mayor información viene recogida en el anexo 2 la configuración completa de todos ellos y en el manual facilitado por Telegesis [27] se puede consultar la funcionalidad de cada registro S.

Entre los cambios que hemos ejecutado está el permitir en ambos dispositivos que puedan recibir mensajes de endpoints diferentes a los proporcionados por Telegesis. Poder configurar la tasa de transmisión a la que queremos trabajar, en nuestro caso lo fijaremos en 115200 baudios. También podemos configurar a través de los registros S40, S42 y S44 la dirección a la que queremos que se envíe un mensaje, determinando los endpoint de salida y entrada, el ClusterID y el ProfileID respectivamente [30]. Además será necesario indicar que el dispositivo soporta tanto el perfil médico como todos los clusters que lo componen a través de los registros del S48 al S4C, como explicamos en el apartado 3.1.4.

Por otro lado, en cuanto a las configuraciones individuales, tenemos que configurar el router para que pueda actuar como un sumidero para los datos del sensor. También es importante definir qué tipo de dispositivo se trata el agente, en nuestro caso un termómetro (1008) en el registro S49. Mientras que en el caso del coordinador, le tenemos que indicar que actúe como un coordinador dándole valores tanto a la dirección de la red (PAN ID) como una contraseña a dicha red.

Además de estas configuraciones existen otros cambios más técnicos que se pueden encontrar en el anexo 2.

4.1.1.2 Automatización de la gestión de red

Una vez llevadas a cabo todas las configuraciones de los registros pasamos a estudiar cómo se gestionan las acciones dentro de una red ZigBee.

En primer lugar recogemos un informe de todo el proceso que hay que seguir establecer la comunicación, para ello se utilizó una interfaz gráfica y otras herramientas derivadas del Zeta Project [29] que nos permiten gestionar los comandos AT.

Una vez que quedo claro el camino a seguir se pasó a desarrollar un

software que gestionara todo el proceso de un modo automático y establece la comunicación entre los diferentes endpoints de los dispositivos dejando la red preparada para soportar los intercambios de tramas en los niveles superiores.

El diagrama de flujo del sistema se presenta en la figura 4.4 y se comporta de la siguiente forma.

En primer lugar se conecta el dispositivo y se abre el puerto de este modo queda publicado un driver a partir del cual comenzaremos a trabajar. Acto seguido comprobamos de qué dispositivo se trata, ya que puede ser el dispositivo que implemente el router o el coordinador, en ambos casos la primera acción que se llevará a cabo será escribir todos los registros que se indican en el punto anterior en función del dispositivo que sea. En el caso de que se trate del coordinador comenzará con la creación de la red y un sistema de escaneo constante para que nuevos dispositivos puedan unirse a ella. Posteriormente, para cada nuevo dispositivo analizaremos sus endpoints buscando que alguno contenga el perfil público ZHC. Si alguno de los dispositivos contiene dicho perfil se registra como un agente y además se establece un sistema de escucha para gestionar los mensajes recibidos de dicho perfil. A partir de un identificador que posee cada agente, podremos saber de qué tipo de dispositivo médico se trata, en nuestro caso tendrá el valor 1008 correspondiente a un termómetro.

Por otro lado, en el sistema que gestiona el router también analizaremos los endpoints que componen los dispositivos ZigBee que se unen a la red con el fin de establecer la comunicación con los mensajes provenientes del coordinador ya que más adelante éste implementará el manager. En este punto también se establece la comunicación con el sensor para implementar el agente pero esto resulta menos relevante en el desarrollo del proyecto (Anexo 4).

Además se establecerá un sistema que envíe un mensaje de prueba cada cierto tiempo para comprobar que la comunicación con el manager se mantiene.

Finalmente quedará un esquema de comunicación como el que se recoge en la figura 4.3.

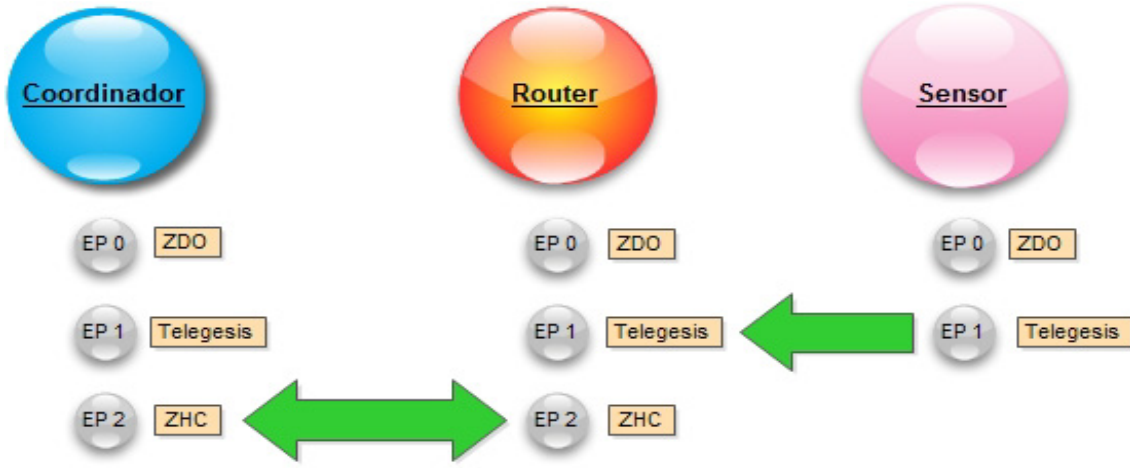


Figura 4.3- Esquema de la comunicación inalámbrica

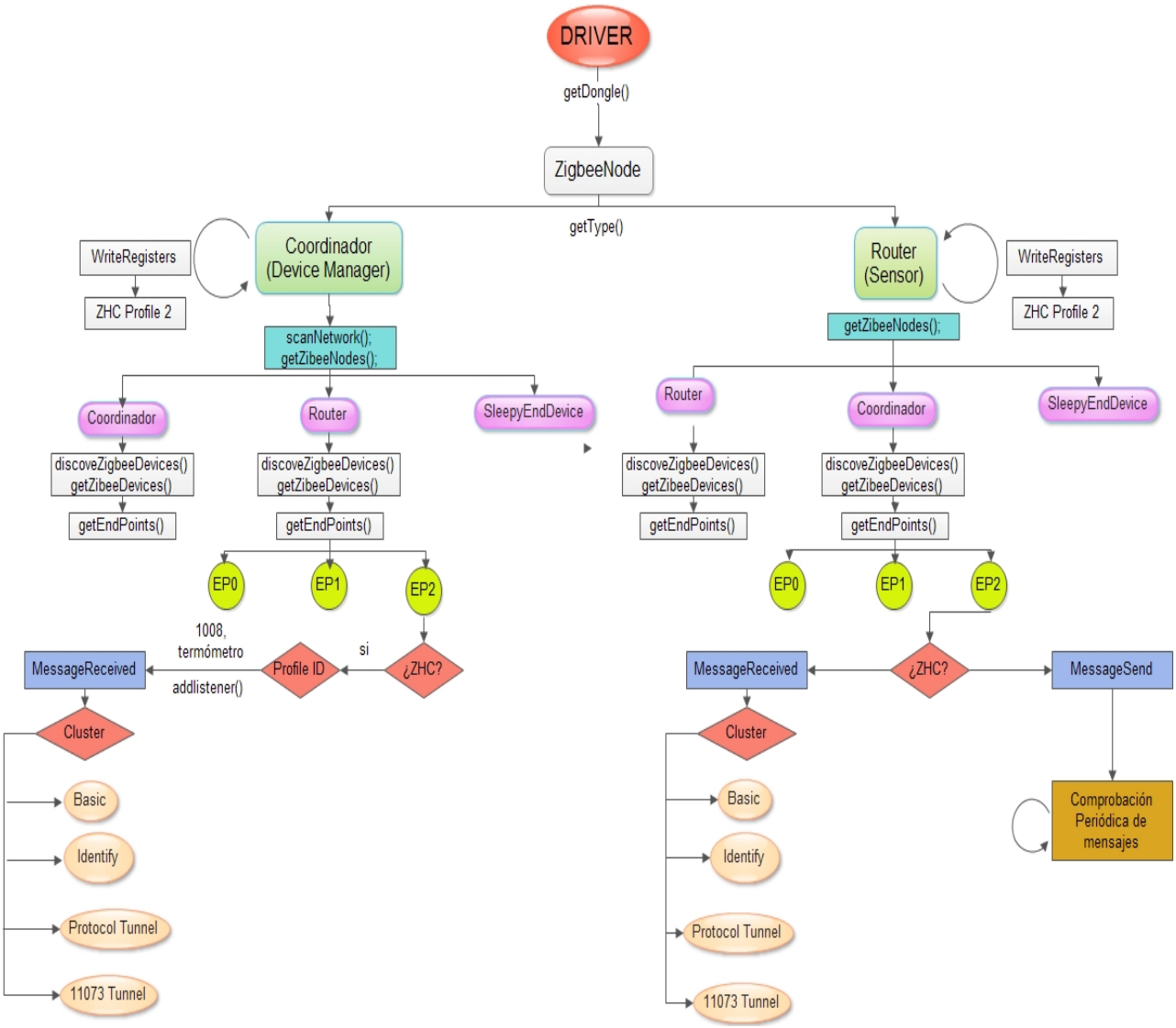


Figura 4.4 Diagrama de flujo de la arquitectura ZigBee utilizada

4.1.2 Implementación del perfil ZHC

Una vez establecidas las conexiones entre los diferentes dispositivos, como se muestra en la figura 4.3, ya estamos en disposición de gestionar los mensajes provenientes del perfil ZHC en ambos dispositivos y a su vez esta implementado el emulador del agente. El siguiente paso es llevar a cabo todo el desarrollo necesario para implementar el perfil médico ZHC en ambos extremos de la comunicación. Por lo tanto deberemos de tener en cuenta que al tratarse de un perfil público el envío de tramas tiene que hacerse de acuerdo a las especificaciones recogidas en la ZCL [8], permitiéndonos así el incluir junto el mensaje, el perfil y el cluster al que va dirigido u otra información como un sistema de control, o el comando que va a emplear dentro del cluster.

Este tipo de tramas están compuesta por una cabecera y una secuencia de datos, como se puede observar en las figuras 4.3 y 4.4 y todos ellas van dirigidas a diferentes clusters del perfil 0108.

Bits: 8	0/16	8	8	Variable
Frame control	Manufacturer code	Transaction sequence number	Command identifier	Frame payload
ZCL header				ZCL payload

Figura 4.3 Formato General de una trama ZCL

Bits: 0-1	2	3	4	5-7
Frame type	Manufacturer specific	Direction	Disable default response	Reserved

Figura 4.4 Frame control

Tanto en el apartado 3.4 como en la especificación del perfil [1] se explican que clusters se deben de soportar a ambos lados de la comunicación.

Vamos a prestar especial atención en el *11073 Protocol Tunnel* [7] ya que nos proporciona los atributos necesarios para poder establecer un canal donde se puedan realizar un intercambio de tramas a nivel X73, los tipos de tramas necesarios en este perfil se recogen en la tabla 4.3. Mientras que el procedimiento a seguir para establecer la comunicación es el siguiente, en primer lugar el agente envía solicitudes de conexión al manager periódicamente hasta que recibe una respuesta indicándole que este dispositivo está autorizado para establecer la comunicación y cuando este la recibe ya puede comenzar con la transmisión de las tramas APDU que son las que establezcan la comunicación al nivel del X73, finalmente cuando la comunicación se cierre, se cerrará también el canal enviando una solicitud de desconexión, este proceso queda plasmado en el diagrama de la figura 4.5.

Command identifier field value	Description	Mandatory/Optional
0x00	Transfer APDU	M
0x01	Connect request	O
0x02	Disconnect request	O
0x03	Connect status notification	O
0x04 – 0xff	Reserved	-

Tabla 4.3- Comandos del perfil ZHC

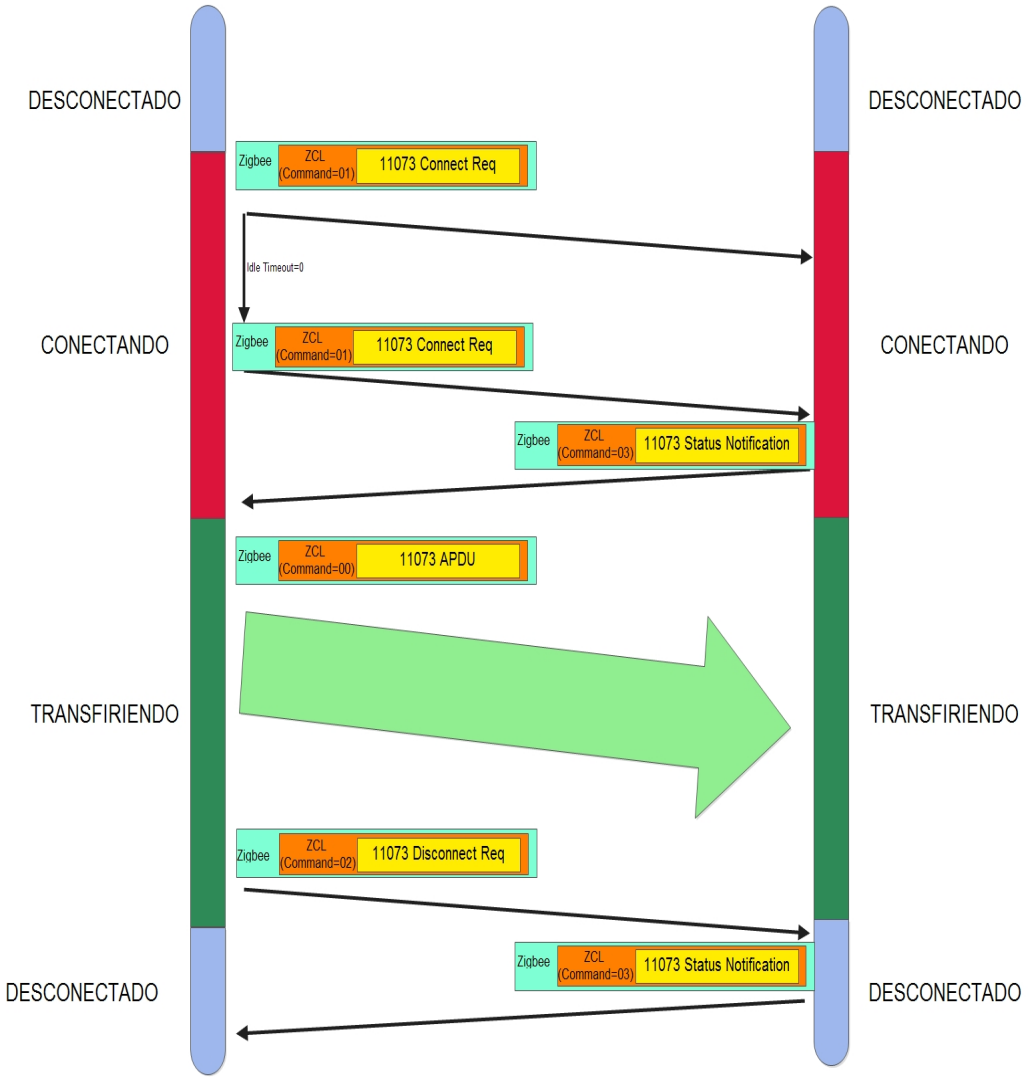


Figura 4.5 Establecimiento del canal ZHC

En el caso de la *petición de conexión* se compone por el identificador de red, el Idle timeout que nos indica el tiempo que pasa hasta que repetimos otra petición de conexión y el número de endpoint de donde procede la petición, como se muestra en la figura 4.6.

Octets	1	2	8	1
Data Type	8-bit bitmap	16-bit unsigned integer	IEEE Address	8-bit unsigned integer
Field Name	Connect control	Idle timeout	Manager target	Manager endpoint

Figura 4.6 Formato de la petición de conexión

La respuesta a dicha petición de conexión, será un campo de ocho bits que tomará uno de los valores mostrados en la tabla 4.4, en el caso de recibir una respuesta satisfactoria, que sería un valor 0x01 ya estaríamos en disposición de comenzar a ejecutar los pasos necesarios para establecer el protocolo x73.

Valor	Designación	Descripción
0x00	Disconnected	Indica que el agente ha sido desconectado del tunel
0x01	Connected	Indica que el agente se ha conectado al tunel
0x02	Not_autorized	Indica que esa petición no a sido autorizada por el momento
0x03	Reconnect_request	El agente necesitaría realizar otra conexión con el manager
0x04	Already_connected	Indica que la petición de conexión ha sido rechazada porque ya existe una conexión entre ambos.

Tabla 4.4 Estados de respuesta a la petición de conexión

Una vez establecida la comunicación en cualquier momento podemos recibir una *petición de desconexión* que al igual que la respuesta a la petición de conexión, contiene un campo de 8 bits (figura 4.7) cerrando de este modo el tunel de comunicación y recibiendo una respuesta con el valor 0x00.

Octets	8
Data Type	IEEE Address
Field Name	Manager IEEE address

Figura 4.7 Formato de trama de la petición de desconexión

4.1.3 Implementación del protocolo 11073

Llegados al punto en el que nuestro sistema ya es capaz de crear y gestionar la red, ambos dispositivos han sido configurados correctamente y se ha establecido un canal de comunicaciones, ha llegado el momento de implementar el protocolo 11073.

Este estándar como se explica en el apartado 3.2 se gestiona a través de una máquina de estado finitos que implementa los estados desconectado, conectado, desasociado, asociado, configurando y se comporta del modo que se muestra en el esquema de la figura 4.8 [31].

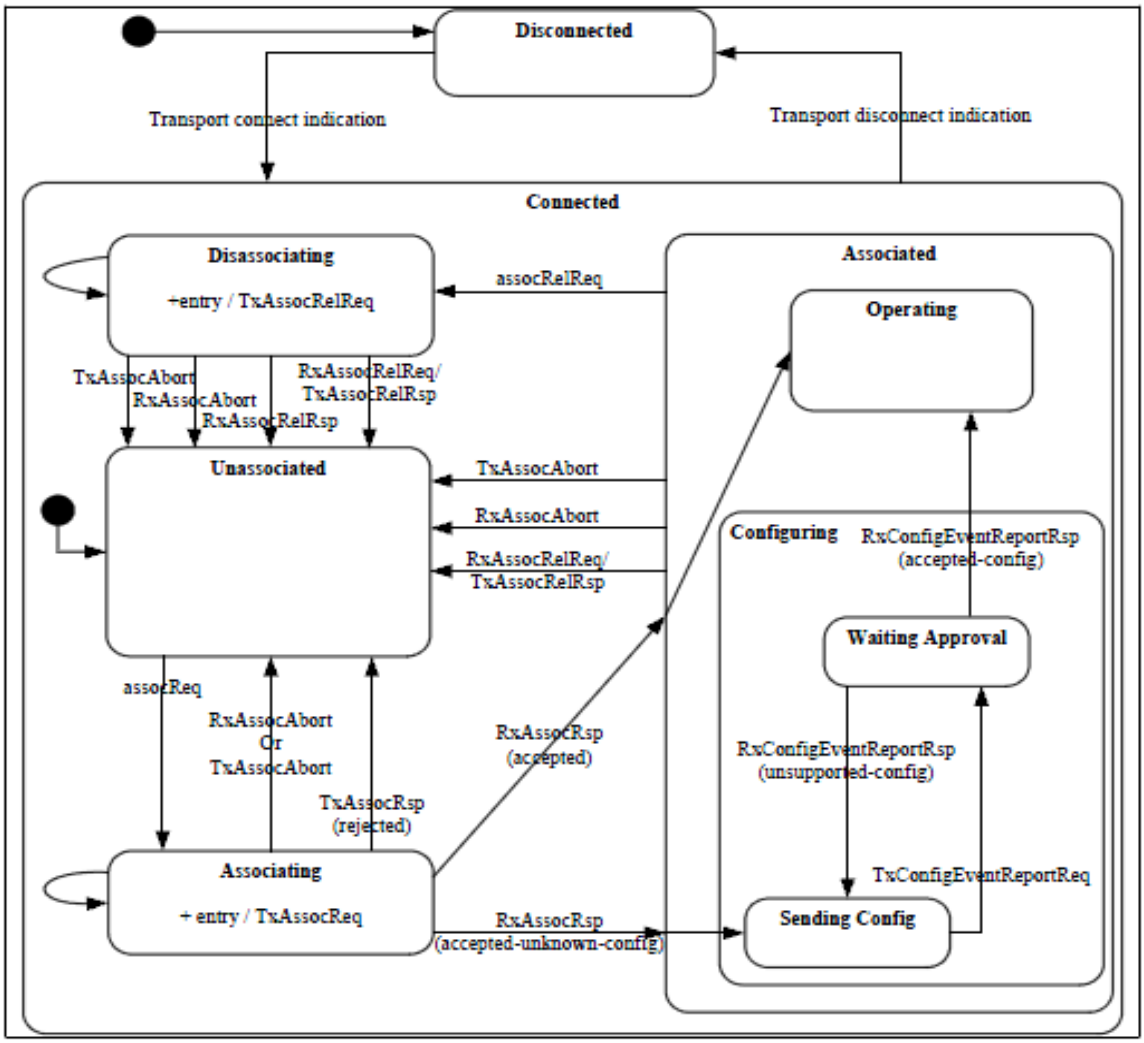


Figura 4.8 – Estados de protocolo 11073

Para poder llegar a implementar esta parte del proyecto sobre nuestro código ha sido necesario adaptar el trabajo de otro proyecto fin de carrera este proyecto fue elaborado en julio de 2010 y tiene el nombre de *"Optimización de una Plataforma Telemática para Monitorización de Pacientes orientada a u-Salud, y basada en Estándares y Plug-and-Play"* , continua con el trabajo de una plataforma que ayudo a desarrollar el grupo X73Spain [4] en la que se implementaba el protocolo 11073 a través de la tecnología Bluetooth, sobre un sistema operativo android.

Lo que obtendremos una vez adaptado el código serán las herramientas necesarias para poder gestionar toda la comunicación X73.Permitiendonos la posibilidad de generar las tramas APDU, en función de la necesidad que necesitemos en cada momento, además de todas las herramientas necesarias para poder analizar y crear todas las peticiones, así como un sistema de codificación y decodificación basado en el protocolo MDER (Medical device encoding rules) que aplicaremos sobre las tramas APDU antes de enviarlas a través del canal [16].

Estos "bloques de herramientas" vendrán implementados en dos bundles, uno correspondiente al agente y otro correspondiente al manager, y sobre los que se apoyará tanto router como coordinador respectivamente para gestionar dicho protocolo como se muestra en la figura 4.9.

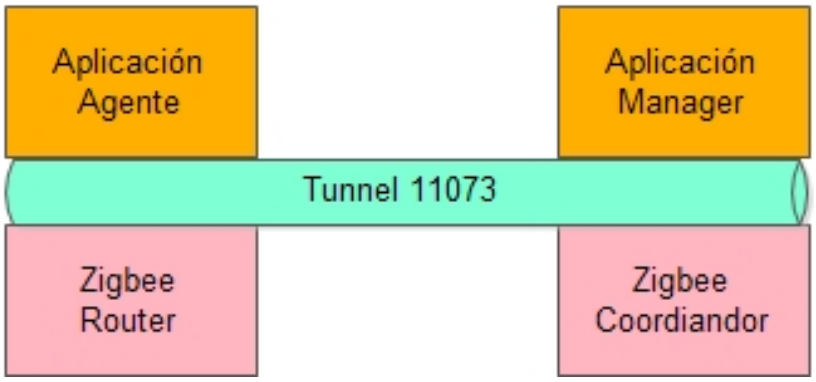


Figura 4.9 Diagrama del protocol tunel

El sistema funciona del siguiente modo, tanto manager como agente parte de un estado de desconexión, en cuanto el agente detecta que se ha creado el canal ZHC a través de éste le envía una *Association Request* y tras los trámites correspondientes pasan al estado asociados. Una vez asociados el manager le solicita al agente su configuración para posteriormente alcanzar el estado operando, donde el router comenzará a introducir los datos provenientes del sensor y encapsularlos en tramas APDU para enviárselas al coordinador. Estando en el estado operando en cualquier momento se puede pasar al estado desasociado bien sea por un error o bien porque se ha finalizado el envío de medidas.

Finalmente, el formato que tendrá el dato que enviamos una vez que hemos alcanzado el estado operando, será como se muestra en la Figura 4.10

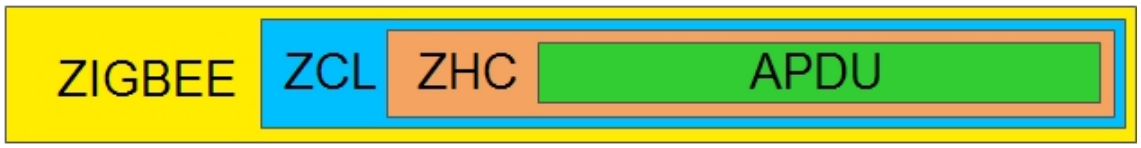


Figura 4.10 Formato del dato transmitido

Por lo tanto el dato de temperatura se codificará dentro de una trama APDU, que a su vez irá codificada con el protocolo MDER. Se envía a través del canal ZHC con el comando 00 (indica que se trata de una trama APDU) y a su vez encapsulada en una trama ZCL. Esta trama en forma de un conjunto de bytes se enviará al perfil médico del manager, para que ésta pueda ser desencapsulada y decodificada, obteniendo finalmente el dato proveniente del sensor.

5 Desarrollo e implementación

Una vez comprendida la arquitectura seguida, en este apartado se presenta un estudio más detallado a nivel de software. Este punto parte del entorno de trabajo que se ha utilizado y hace especial hincapié en determinadas partes del código que se ha desarrollado.

5.1 Entorno de trabajo y desarrollo previo

En primer lugar indicar que el proyecto se ha desarrollado en un entorno Java y con el fin de poder desarrollar aplicaciones en forma de módulos independientes vamos a utilizar la especificación OSGi [33] que aparte de esto también permite que estos módulos puedan colaborar entre ellos. Por otro lado se ha utilizado una implementación de código abierto llamada Felix [34], que se trata de un contendor que nos proporciona Apache para la implementación de OSGi. Como entorno de trabajo se ha utilizado Karaf [35], el cual nos proporciona un framework con muchas posibilidades y amplia flexibilidad. En la Figura 6 se muestra una imagen del aspecto de la consola de Karaf.

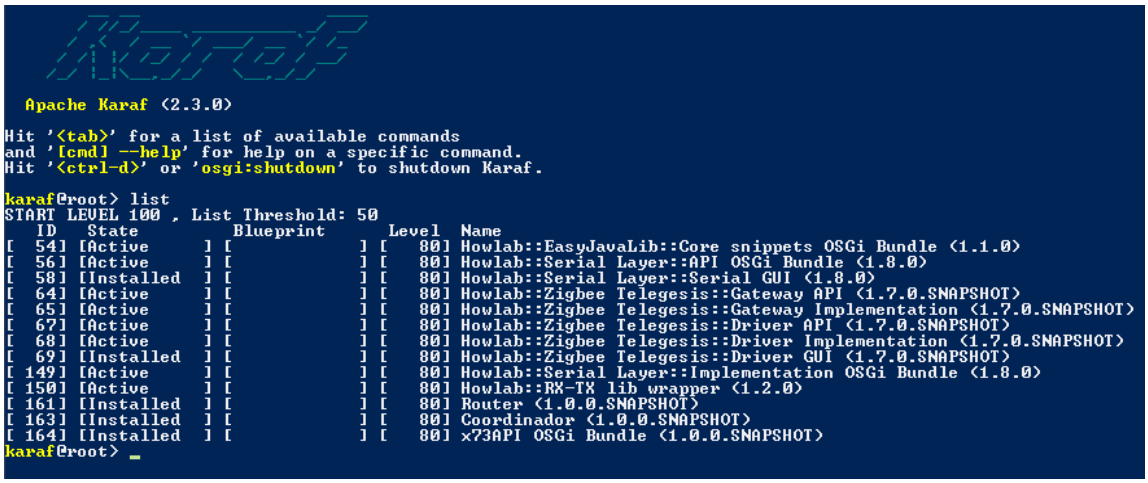


Figura 5.1.Consola Karaf

Una vez presentado el entorno de desarrollo, para poder comenzar a trabajar es necesario instalar el driver de Telegesis [1], para que el ordenador pueda reconocer el modulo. Por otro lado para poder obtener las herramientas necesarias que nos permitan trabajar con los dispositivos ZigBee, partiremos del código abierto implementado por Howlab.

Para poder emplearlo es necesario instalar los siguientes bundles en nuestro repositorio [36]:

```
es.unizar.howlab.easyjavalib
es.unizar.howlab.thirdparty
es.unizar.howlab.core.io.serial-api
es.unizar.howlab.core.io.serial-impl
es.unizar.howlab.core.zigbee.telegesis-gateway api
es.unizar.howlab.core.zigbee.telegesis-gateway impl
es.unizar.howlab.core.zigbee.telegesis-driver api
es.unizar.howlab.core.zigbee.telegesis-driver impl
```

Estos bundles son parte del proyecto Zeta Framework el cual nos proporciona una plataforma para poder interactuar con los dispositivos ZigBee, pudiendo así desarrollar nuestra aplicación a partir de aquí. Como observamos en la figura 5.2 nos proporciona una interfaz gráfica que nos permite realizar operaciones básicas de un modo manual, así como la lectura y escritura de los registros S.

Esta interfaz gráfica resultará de gran utilidad ya que es aquí donde se han realizado todas las pruebas para comprender y estudiar el correcto funcionamiento de la tecnología ZigBee, la creación de la red, el envío de mensajes, la configuración de clusters y endpoints, es decir la fase de aprendizaje de todos los pasos que tendremos que implementar posteriormente de un modo automático.

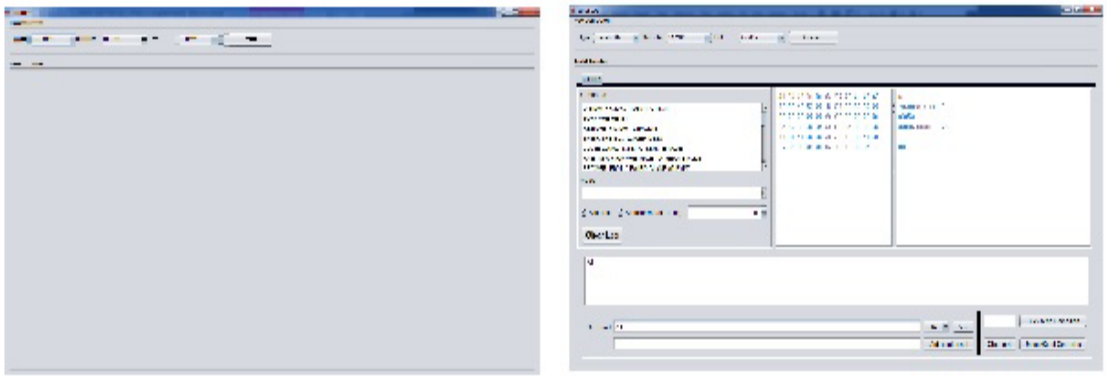


Figura 5.2.Interfaz gráfica serial Gui desconectado y conectado

Debido a que por la arquitectura seguida en el proyecto se lleva un estudio paralelo de ambos lados de la comunicación tanto en el agente como en el manager, resulta necesario trabajar desde dos puntos de conexión diferentes y puesto que trabajar desde dos ordenadores diferentes no resulta para nada practico, es aquí donde Karaf nos presenta una perfecta solución ya que existe la posibilidad de abrir dos instancias diferentes para poder trabajar de un modo completamente independiente (Figura 5.3).

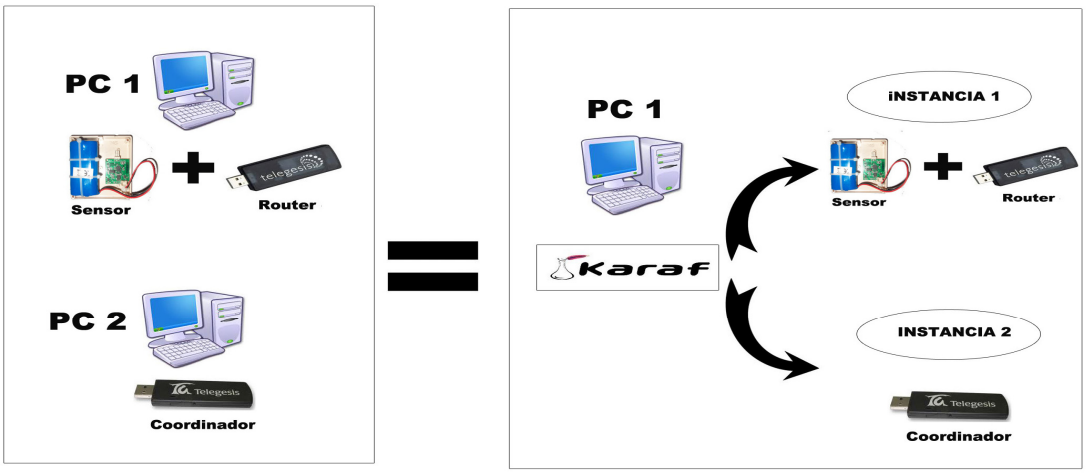


Figura 5.3 Dos instancias de Karaf

Antes de adentrarnos a analizar el código también puede resultar de utilidad explicar la estructura que siguen los proyectos que hemos usado, estos proyectos se tratan de proyectos Maven [37]. Los proyectos de este tipo se distribuyen en varias carpetas por un lado tenemos las fuentes que componen el bundle, que se trata de una serie de módulos independientes pero que interaccionan entre si durante la ejecución. Mientras que por otro lado se encuentran las dependencias que tiene el proyecto que pueden ser o bien otros proyectos que tengamos abiertos o de servicios que se encuentren en la web. Otro de los archivos importantes a la hora de definir un proyecto Maven es el Project Object Model (POM) que sirve para describir el proyecto de software a construir, sus dependencias con otros módulos y componentes externos, así como el orden de construcción de los elementos.

El esquema que nos quedaría al crear un proyecto es el que se puede ver en la figura 5.3.

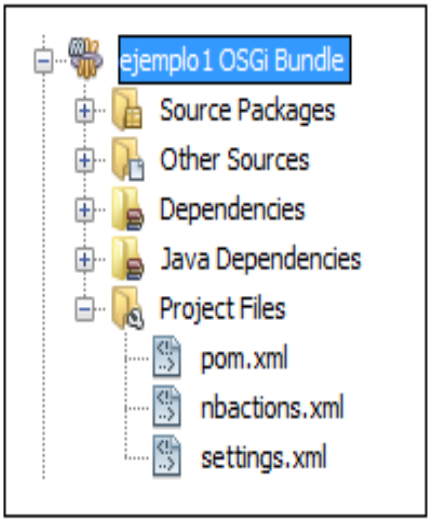


Figura 5.3 Modelo de proyecto Maven: OSGi bundle

Una vez expuestas las herramientas que vamos a utilizar pasamos a analizar de un modo más detallado el código que hemos implementado.

5.2 Estructura del código

En este apartado se van a presentar los diferentes bloques que componen el proyecto, como ya hemos ido explicando en apartado anteriores se van a desarrollar dos módulos principales, uno será el encargado de gestionar toda la parte correspondiente al coordinador de la red y sobre el montar la estructura del manager. Mientras que el otro bloque será el encargado de gestionar al agente. Ambos bloques se apoyarán en otros dos bloques X73APIManager y X73APIAgente que serán los encargados de proporcionarle todas las herramientas necesarias para gestionar el protocolo X73.

5.2.1 Coordinador

En primer lugar vamos a explicar cómo se ha desarrollado el módulo que gestiona el coordinador, en la figura 5.4 se muestra un esquema de los bloques que lo componen quedando su funciones definidas en cuatro bloques que son por un lado las funciones que desempeña dentro de la red ZigBee, por otro lado la implantación tanto del perfil ZHC como del protocolo X73 esta última apoyada como indicamos en el apartado anterior en otro bloque llamado X73API MANAGER y de un modo paralelo a los anteriores, se desarrolla la interfaz gráfica para mostrar al usuario en todo momento lo que está ocurriendo además de mostrar por pantalla el dato de temperatura obtenido por el sensor.

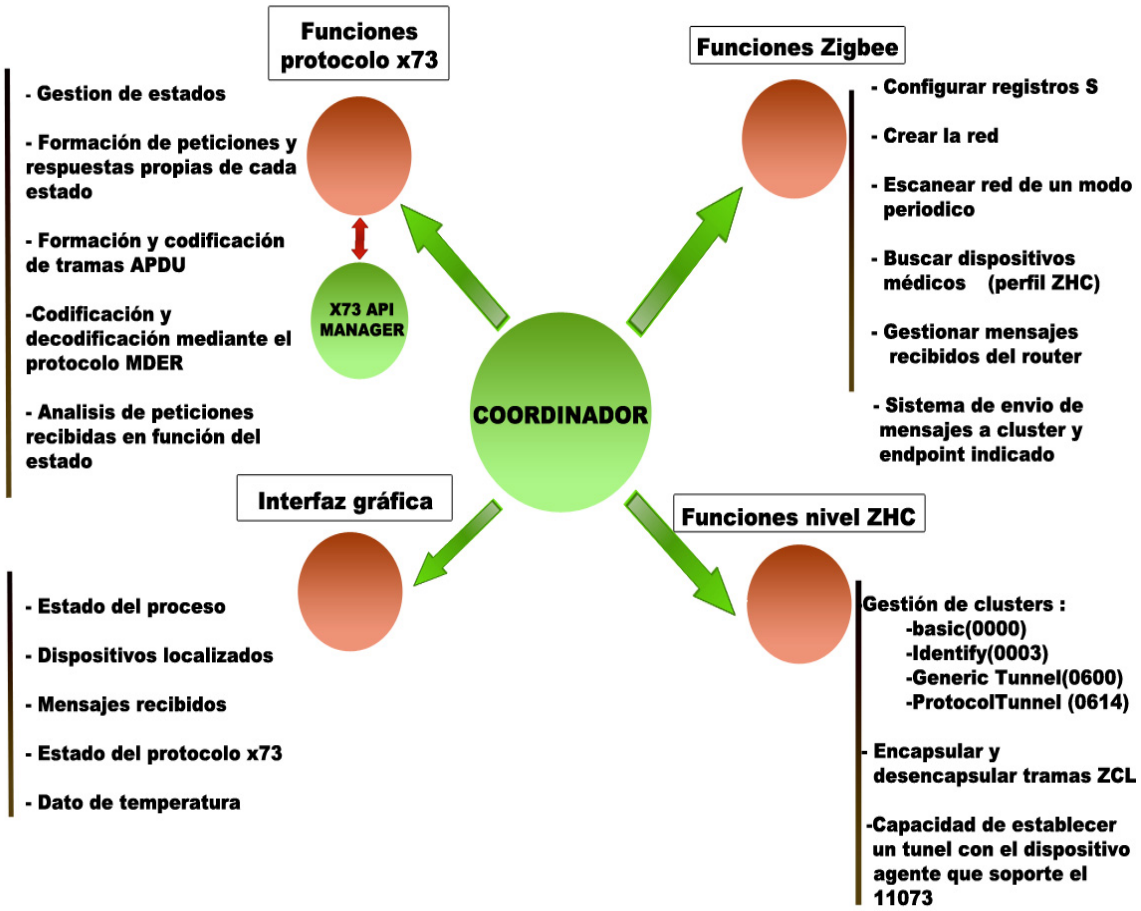


Figura 5.4 Esquema del módulo del coordinador

En primer lugar las funciones que desempeña a nivel de ZigBee comienzan con la configuración de todos los registros indicados en el punto 4.1.1.1, posteriormente pasa a la creación de la red y a escanear periódicamente dicha red con el objetivo de detectar si algún nuevo nodo se ha unido a la red, en cuanto descubre un nuevo nodo pasa a analizar sus endpoints y los compara con el perfil médico (0108), en el caso de que alguno de ellos lo contenga, lo almacena e n una lista de agentes. Posteriormente de dicha lista de agentes, localizamos el que su identificador Device ID tome el valor 1008 ya que esto nos indicará que se trata de un termómetro, es entonces cuando se establece un sistema de escucha para poder gestionar los mensajes recibidos desde dicho endpoint. Llegados a este punto se podría decir que la comunicación a nivel ZigBee que va desde el agente al manager

ya está establecida. Una parte del código que gestiona el descubrimiento de un nuevo nodo lo encontramos en la figura 5.5.

Para poder procesar los mensajes procedentes del agente dirigidos al perfil ZHC, el dispositivo que implementa el manager, tendrá que ser capaz de en primer lugar desencapsular una trama ZCL obteniendo los diferentes campos que la componen (figura 4.3), y tramitarlo en función del cluster al que vaya dirigido. Al implementar el lado correspondiente al cliente, solo será necesario que implemente el ProtocolTunel 11073 ="0614" como se ha indicado en la Tabla 3.4.

```
if (Nodos.containsKey(nodeAddress)) {
    i++;
}
else{
 Nodos.put(nodeAddress, nodos[i]);
 System.out.println("Se ha registrado el nodo : "+nodeAddress);
 IntGrafica.EscribirEUI1(nodeAddress, nodos[i].getType().toString());
 System.out.println("");System.out.println("El nodo es:"+nodos[i]+" y es un "+nodos[i].getType());
 nodos[i].discoverZigbeeDevices();
 devices=nodos[i].getZigbeeDevices();
 int j=0;
 while (j<devices.length){
  if (ZDevices.containsKey(devices[j])) {}else{
      String deviceAddress=devices[j].toString();
      ZDevices.put(deviceAddress, devices[j]);
      System.out.println("Se ha registrado el device : "+deviceAddress);}
   IntGrafica.CambiarColorZHC();
   if (devices[j].getEndPoint()!=0){
      devices[j].discover();
      System.out.println("       El device "+j+" del "+nodos[i].getType()+
      " tiene el EP "+devices[j].getEndPoint()+" y el perfil "+devices[j].getProfileId());
      if ("0108".equals(devices[j].getProfileId()) && nodos[i]!=node ){
         ZDagente=devices[j];
         System.out.println("--> ZHC  (Zigbee Health Care) Encontrado <--");  System.out.println("");
         addlistener(devices[j]);
         String DispMed= devices[j].getDeviceId();
         if (DispMed.substring(0,4)=="1008"){DispMed="  Termometro"; }
         ArrayList<Agent> a=ZHCManagedAgents.getAgents();
         System.out.println("");
         IntGrafica.EscribirDispMedico(nodeAddress,DispMed);
         byte[] sms = new byte[]{0,1};
         nodos[i].sendMessage(sms);
```

Figura 5.5 Procedimiento al encontrar un nodo nuevo

Ya que nuestro agente está programado para establecer el tunel ZHC una vez que localice al manager. El manager tiene que estar preparado para llevar a cabo un intercambio de tramas con el agente y tramitarlas en función del comando que las acompañe (Tabla 4.3), en la figura 5.6 se muestra un fragmento del código encargado de analizar los mensajes dirigidos al cluster 0614.

Así pues una vez que el canal se halla establecido correctamente, se pondrá en funcionamiento la máquina de estados que controla el protocolo X73 y se comenzará a implementar el intercambio de tramas APDU hasta que ambos dispositivos realicen todos los ajustes necesarios, como son la asociación de los dispositivos, el envío de solicitudes o el envío de la configuración, para finalmente alcanzar el estado operando [17].

Finalmente al alcanzar el estado operando en ambos extremos de la comunicación, el agente pasará a transmitir de manera periódica el dato de temperatura encapsulado y decodificado dentro del campo PRST de una trama APDU, mientras que en el manager se llevará a cabo el proceso inverso para acabar almacenando dicho dato clínico junto con la fecha y hora en que ha sido tomado en un vector de medidas, para poder efectuar un correcto seguimiento del paciente.

```
if ("0614".equals(clust)){
     switch (command) {
     //00  Received APDU
        case 0x00:
             ApduType apduReceived = new ApduType();
             IDecoder decoder = CoderFactory.getInstance().newDecoder("MDER");
             ByteArrayInputStream is = new ByteArrayInputStream(payload);
             apduReceived = decoder.decode(is, ApduType.class);
             Gestion.ProcAPDU(apduReceived);
             break;
        case 0x01:
            AssociationResponseType AssResponse = null;
            if (estado==EstadosManager.DESCONECTADO){
                AssResponse=AssociationResponseType.CONNECTED;
            }
            if (estado==EstadosManager.CONECTADO){
                 AssResponse=AssociationResponseType.ALREADY_CONNECTED;
            }
             IntGrafica.ColorConectando();
             estado=EstadosManager.CONECTADO;
             System.out.println("Analizando la peticion de conexion estado "+estado)
             byte[] AssociationResp=ManagerZHC.AssociationResponse(zb,AssResponse);
             zb.sendMessage("0108", "0614", AssociationResp);
             break;
```

Figura 5.6 Gestión de una trama en el cluster 0614

Una vez alcanzado el estado operando se continuará con el intercambio de tramas de un modo periódico, hasta que se cierre el canal de comunicación ya sea de un modo voluntario o involuntario, en este caso el estado del manager pasaría a ser desconectado y canal ZHC se cerraría.

De un modo paralelo a todo este proceso se llevará a cabo el desarrollo de una interfaz gráfica que presenta el aspecto de la figura 5.7. Ya que el funcionamiento del sistema trabaja de un modo automático una vez que se abre el puerto, no existe ningún botón de inicio, el programa comenzará a correr, mostrándonos la fase del proceso en la que se encuentra, los nodos que se han registrado y el estado del protocolo X73, para finalmente acabar mostrando el dato de temperatura recogido por el sensor. Además tendremos la posibilidad de observar los datos de temperatura recogidos, para poder llevar a cabo un seguimiento más preciso del paciente.

En el Anexo 2 se encuentra una explicación detallada de cada uno de los bloques que componen esta interfaz.

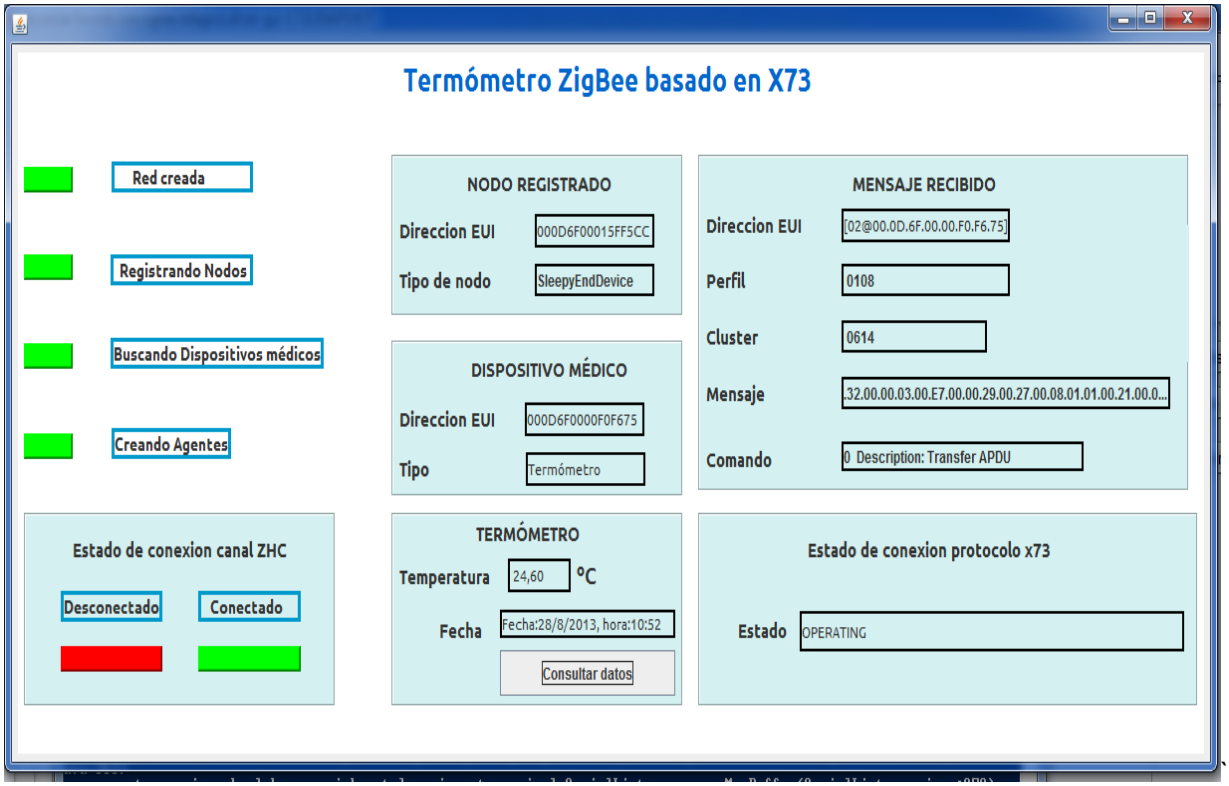


Figura 5.7 Interfaz gráfica

5.2.2 Router

Del mismo modo que hemos explicado el funcionamiento del bloque que gestiona el coordinador vamos a pasar a explicar cómo se ha desarrollado el bloque que implementa el agente, al igual que el coordinador también se puede dividir en varios bloques como se muestra en el esquema de la figura 5.8. Por un lado tenemos la parte que gestiona la conectividad dentro de la red ZigBee y la encargada de crear las conexiones inalámbricas entre los dispositivos y por otro lado al igual que en el coordinador, existen dos bloques encargados de implementar tanto el perfil ZHC como el protocolo 11073, este último apoyándose en el módulo X73API AGENTE. En este caso tenemos que tener en cuenta que el agente será el encargado de realizar todos los trámites necesarios para iniciar el protocolo de comunicaciones, estableciendo en primer lugar el canal ZHC y una vez creado iniciar el protocolo X73.

La parte correspondiente a la gestión del bloque ZigBee tiene una arquitectura similar al caso del coordinador ya que lo primero que hace es configurar el dispositivo ETRX2, escribiendo los registros indicados en el Anexo 2 y posteriormente iniciar los procesos correspondientes a la capa ZigBee. Una vez que se encuentra dentro de la red pasará a analizar los dispositivos que la componen con el objetivo de localizar el coordinador de la red o el manager en el nivel de X73.

Por otro lado, y de un modo ajeno al intercambio de datos médicos, también establece comunicación con el sensor para poder recibir los datos de temperatura y emular de este modo el comportamiento de un agente (Anexo 4).

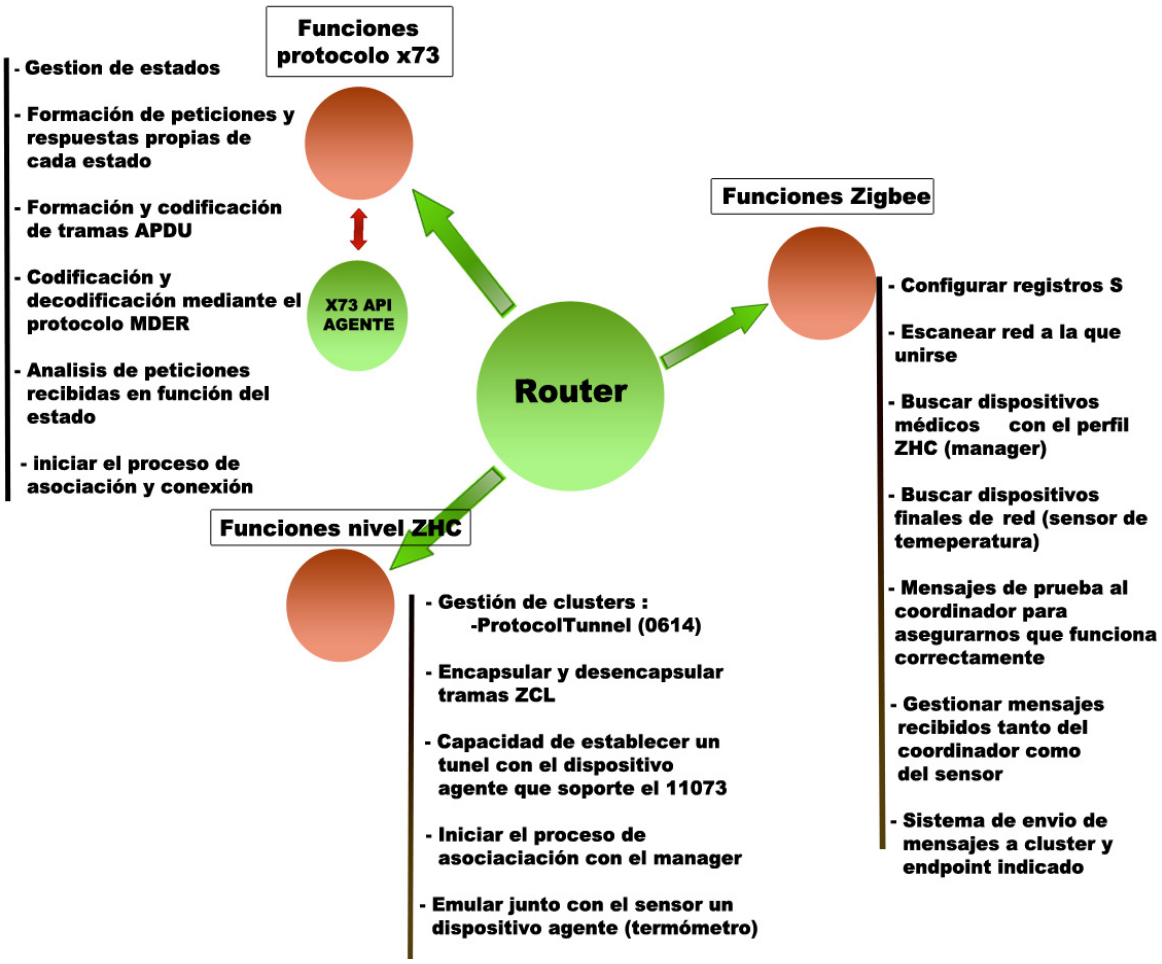


Figura 5.8 Esquema del módulo del router

Una vez localizado el manager, pasará a localizar el endpoint que implemente el manager, que será el que contenga el perfil 0108 y será con este con quien se iniciará de un modo automático el establecimiento del canal ZHC. Por tanto el canal se creará entre el endpoint que implemente el perfil médico dentro del router y el perfil médico del coordinador.

El proceso comenzará enviando una solicitud de asociación dentro de una trama ZCL, con el comando 01 (la forma que se crea esta solicitud se puede comprobar en la figura 5.9) por lo que será necesario implementar el perfil ZHC del modo que se explica en el apartado 3.1.4 y de este modo será capaz de gestionar los datos dirigidos a los siguientes clusters:

- Basic = "0000"
- Identify ="0003"
- GenericTunnel ="0600"
- ProtocolTunel 11073 ="0614"

A su vez también tiene la capacidad de encapsular y desencapsular las tramas ZCL.

Una vez que haya recibido la respuesta aceptando la conexión entrará en funcionamiento el último bloque correspondiente a la norma X73, donde apoyándose en el módulo correspondiente al X73 API AGENTE, gestionará el intercambio de tramas APDU con el manager.

```
public static void SendAssociationRequest(ZigbeeDevice dev) {
          //Creamos la asociacion APDU (Dentro de una trama ZCL)
          if ((estadoZHC!= EstadosAgente.DESCONECTADO)&&(estadoZHC!= EstadosAgente.CONECTANDO)){
               timeout.Stop();
          }else{
          estadoZHC=EstadosAgente.CONECTANDO;
          byte[] apduRequest;
          apduRequest=AgenteZHC.createAssociationRequest(dev,dev.getEndPoint() );
          byte Command= 01;
          byte[] Request;
          Request=ZCLFrame.Compose(apduRequest, dev, Command);
          dev.sendMessage("0108", "0614", Request);
          int cont=(int)apduRequest[2];
          timeout = new IdleTimeout(dev,cont);
          timeout.Start();
          }
}
```

Figura 5.9 Petición de conexión del canal ZHC

El proceso correspondiente al protocolo X73 comenzará enviando una solicitud de asociación [1], codificando una trama APDU dentro de una trama ZCL, como se puede ver en la figura 5.10. De este modo alcanzará el nivel asociado y si el manager lo requiere éste pasará a enviarle su configuración y si es aceptada, ambos dispositivos alcanzarán el estado operando. Es aquí cuando se comenzará a enviar los datos de temperatura encapsulados dentro de una trama APDU y codificados a través del protocolo MDER. Estos datos tendrán que ser enviados a través del canal ZHC, al perfil, clusters y endpoint adecuados, finalmente el dato tendrá un formato como se muestra en la figura 5.11.

Por último, si en cualquier momento ya sea de un modo voluntario o involuntario se cortase la comunicación, ambos dispositivos pasarían a un estado desconectado y se cerraría el canal ZHC.

```
public static void Initialize(ZigbeeDevice device) {
     associated = false;
     measureSentOK = false;
     normalRelease = false;
     Zdevice = device;
     try {
          /** * STATE: CONNECTED - ASSOCIATING **/
          thermID = String.valueOf(R.string.therm_id);
          if( Analizar.estadox73==EstadoX73.DISCONNECTED){
          createAssociationRequest(thermType);
          InternalEventReporter.agentChangeStatus(thermID, "Associating");
          //Send Association Request  -->  ASSOCIATING
          sendApdu(apduAsReq);
          if (associated) {
               InternalEventReporter.agentChangeStatus(thermID, "Operating");
          } else {
               InternalEventReporter.agentConnectionEvent(thermID, connError)
          }
          }} catch (Exception e) {
          System.out.println("Error Thermometer() -- " + e.toString());
          InternalEventReporter.agentDisconnected(thermID, "Disconnected");
     }
}
```

Figura 5.10- Petición de asociación protocolo 11073

Así pues a modo de resumen, la decisión final de diseño que se decidido implementar es por un lado todo el desarrollo correspondiente a la gestión del manager en un endpoint dentro un módulo ETRX2 y por otro lado toda la implementación de la arquitectura que compone el agente.

El dato de temperatura se lo proporcionamos a través de un sensor dotado de un módulo ZigBee aunque este dato de temperatura podría provenir de cualquier otro sensor ya que el grueso del proyecto se encuentra en la implantación de la lógica de gestión que gestiona tanto el agente como el manager, la emulación de cómo se ha formado el agente se explica en el anexo 4.

Una vez que se ha implementado toda la parte correspondiente al agente estaríamos en disposición de trasladar el software diseñado al hardware de un sensor ZigBee para que este pueda actuar por si solo como un agente médico.

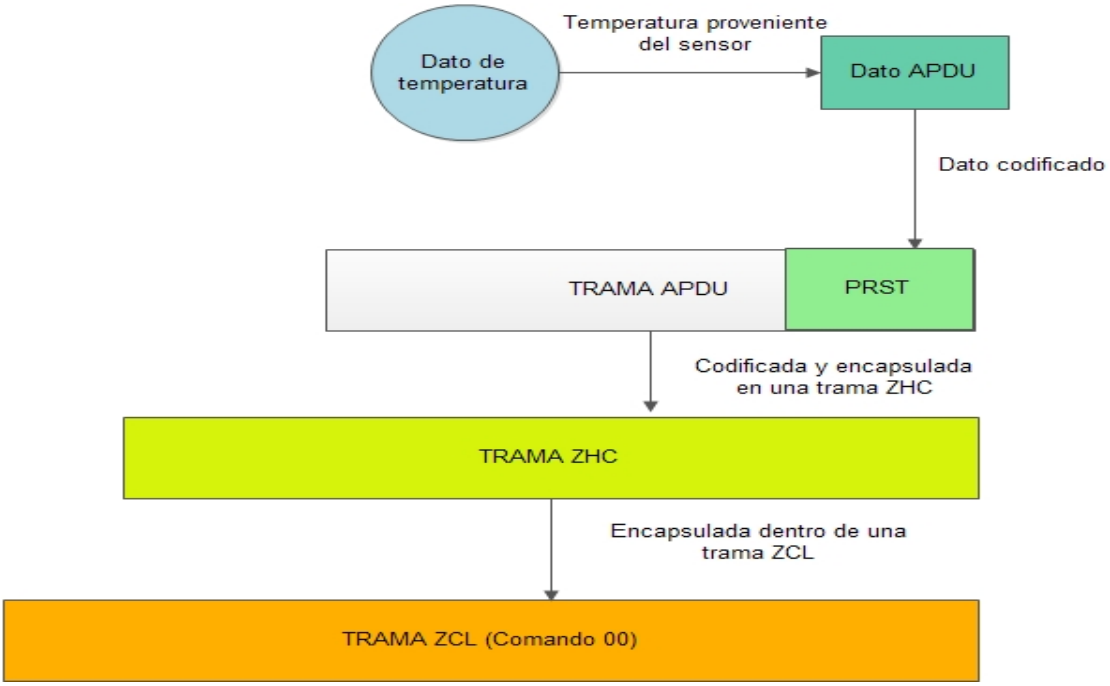


Figura 5.11 Proceso de encapsulado del dato de temperatura

Para la gestión de todo lo expuesto a lo largo de este punto, se ha llevado a cabo a través de dos bloques uno correspondiente a la implementación del manager y otro a la implementación del agente, a su vez cada uno de estos bloques están compuestos por un conjunto de clases y métodos que interactúan entre sí como podemos encontrar en la figura 5.12.

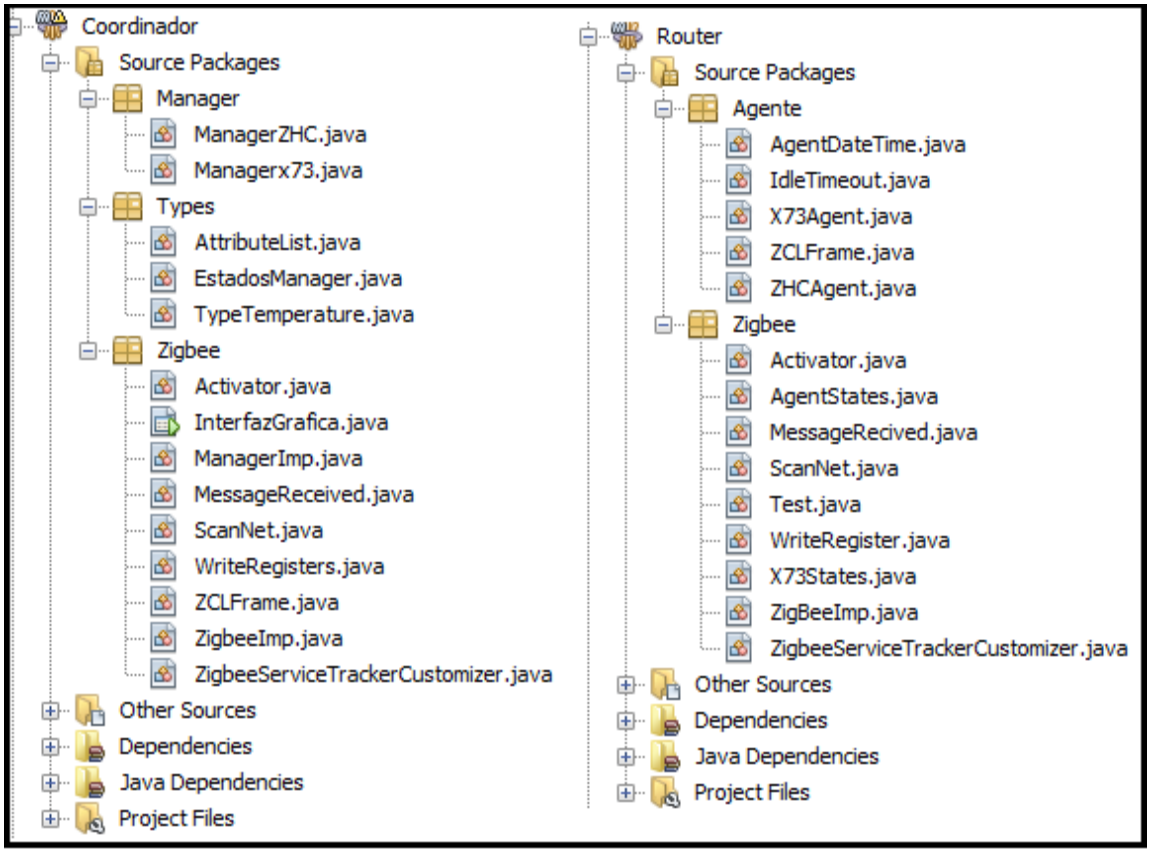


Figura 5.12 Composición correspondiente al Router y al Coordinador.

5.2.3 X73API Manager y X73APIAgente

Estos dos bloques se tratan de los bloques donde se apoyan el coordinador y router respectivamente, para gestionar las herramientas correspondientes al protocolo 11073. Ambos bloques Implementan el código necesario para llevar a cabo tanto la codificación como la decodificación MDER, los estados, definiciones, así como todos los estados de la máquina de estados que implementa el protocolo X73 (Figura 4.8). En la figura 5.13 se muestra un ejemplo de cómo procesa una trama APDU dentro cuando nos encontramos dentro del estado asociando.

No vamos a entrar a detallar estos bloques en detalle ya que como ya hemos explicado no han sido desarrollados para este proyecto y se trata de un sistema correspondiente a otro PFC, aunque sí que es cierto que ha habido que emplear mucho tiempo en modificar el código para que pudiera ser compatible con nuestro sistema. Las principales modificaciones que se han llevado a cabo han sido por un lado, adaptar todas las partes que hacían referencia a programación android para ajustarla al lenguaje Java y por otro lado también ha sido necesario cambiar toda la parte del envío de mensajes ya que este proyecto fue diseñado para trabajar con tecnología Bluetooth en lugar de tecnología ZigBee.

```
public synchronized void process(ApduType apdu) {
        if (apdu.isAarqSelected()){
                associating(apdu.getAarq());
                state_handler.changeState(new CheckingConfig(state_handler));
        } else if (apdu.isAareSelected() || apdu.isRlrqSelected() || apdu.isPrstSelected()){
                state_handler.send(MessageFactory.AbrtApdu_UNDEFINED());
                state_handler.changeState(new MUnassociated(state_handler));
        }
}

@SuppressWarnings("unchecked")
@Override
public synchronized boolean processEvent(Event event) {
        if (event.getTypeOfEvent() == EventType.REC_CORRUPTED_APDU) {
                state_handler.send(MessageFactory.AbrtApdu_UNDEFINED());
                return true;
        } else if (event.getTypeOfEvent() == EventType.IND_TRANS_DESC) {
                Logging.error("2.2) IND Transport disconnect. Should indicate to application layer...");
                state_handler.changeState(new MDisconnected(state_handler));
                try {
                        ExternalEvent<Boolean, Object> eevent = (ExternalEvent<Boolean, Object> ) event;
                        eevent.processed(true, ErrorCodes.NO_ERROR);
                } catch (ClassCastException e) {
```

Figura 5.13- Procesamiento de una trama APDU dentro del estado Asociando

6 Resultados

En este apartado se explican los pasos seguidos para comprobar el correcto funcionamiento del sistema. Se comprueba que se respeten todas fases del proceso de establecimiento del canal, así como comprobar que los datos se envíen correctamente encapsulados en las tramas que les corresponde. A su vez nos cercioramos de que el resultado final resulta satisfactorio y exponemos determinados problemas con los que nos hemos enfrentado.

6.1 Pruebas de software

A lo largo del todo el proyecto se han ido realizando las comprobaciones de que cada fase se completaba satisfactoriamente, tanto la parte correspondiente a la tecnología ZigBee, la implementación del perfil ZHC y finalmente la parte correspondiente a la gestión del protocolo 11073.

Finalmente hemos obtenido un sistema capaz de gestionarse de un modo automático en ambos extremos de la comunicación, capaz de crear un canal ZHC sobre una red ZigBee e implementar sobre dicho canal el protocolo 11073 con el fin de transmitir el dato clínico de acuerdo a los estándares médicos.

Para realizar las comprobaciones necesarias, resulta indispensable contar con dos puntos de acceso. En nuestro caso, como se explica en el apartado 5.1, en lugar de contar con un punto de conexión en ordenadores diferentes, crearemos la plataforma en el mismo ordenador, ya que Karaf nos proporciona la posibilidad de abrir dos instancias completamente independientes entre sí. Para llevar a cabo el control de todas las acciones que se dan a lo largo del proceso tendremos por un lado la parte correspondiente al manager que la evaluaremos a partir de la interfaz gráfica, y por otro lado la parte del router que se evaluará a través de salidas por pantalla que podremos visualizar a través de la consola de Karaf.

Una vez definidas las instancias será necesario en primer lugar instalar todos los bundles necesarios para poder abrir el puerto del dispositivo de Telegesis (apartado 5.1) y posteriormente en un punto de conexión colocar tanto el bundle encargado de gestionar el coordinador y el X73API Manager que es el encargado de suministrarle las herramientas necesarias para gestionar el protocolo X73, mientras que en la otra instancia habrá que hacer lo propio con el router. Los bundles empleados en cada instancia los podemos comprobar en la figura 6.1.

Para abrir el puerto utilizaremos el Serial Gui (Figura 6.2) que nos ofrece la posibilidad de elegir la velocidad a la que queremos que trabaje, en nuestro caso lo iniciaremos con una tasa de baudios de 115200, ya que lo hemos configurado para que trabaje a esta velocidad, aunque sería posible cambiarlo, a través de esta pantalla también es posible comprobar las instrucciones que se envían al módulo ZigBee en todo momento.

Así pues una vez presentados los pasos previos a iniciar el proceso, pasamos a describir cómo funciona finalmente el sistema que hemos desarrollado.

```
karaf@root> list
START LEVEL 100 , List Threshold: 50
   ID   State         Blueprint      Level  Name
[  54] [Active     ] [            ] [   80] Howlab::EasyJavaLib::Core snippets OSGi Bundle (1.1.0)
[  56] [Active     ] [            ] [   80] Howlab::Serial Layer::API OSGi Bundle (1.8.0)
[  58] [Active     ] [            ] [   80] Howlab::Serial Layer::Serial GUI (1.8.0)
[  64] [Active     ] [            ] [   80] Howlab::Zigbee Telegesis::Gateway API (1.7.0.SNAPSHOT)
[  65] [Active     ] [            ] [   80] Howlab::Zigbee Telegesis::Gateway Implementation (1.7.0.SNAPSHOT)
[  67] [Active     ] [            ] [   80] Howlab::Zigbee Telegesis::Driver API (1.7.0.SNAPSHOT)
[  68] [Active     ] [            ] [   80] Howlab::Zigbee Telegesis::Driver Implementation (1.7.0.SNAPSHOT)
[ 149] [Active     ] [            ] [   80] Howlab::Serial Layer::Implementation OSGi Bundle (1.8.0)
[ 150] [Resolved   ] [            ] [   80] Howlab::RX-TX lib wrapper (1.2.0)
[ 166] [Installed  ] [            ] [   80] Router (1.0.0.SNAPSHOT)
[ 167] [Installed  ] [            ] [   80] x73APIAgent OSGi Bundle (1.0.0.SNAPSHOT)
```

```
START LEVEL 100 , List Threshold: 50
   ID   State         Blueprint      Level  Name
[  54] [Active     ] [            ] [   80] Howlab::EasyJavaLib::Core snippets OSGi Bundle (1.1.0)
[  56] [Active     ] [            ] [   80] Howlab::Serial Layer::API OSGi Bundle (1.8.0)
[  58] [Active     ] [            ] [   80] Howlab::Serial Layer::Serial GUI (1.8.0)
[  64] [Active     ] [            ] [   80] Howlab::Zigbee Telegesis::Gateway API (1.7.0.SNAPSHOT)
[  65] [Active     ] [            ] [   80] Howlab::Zigbee Telegesis::Gateway Implementation (1.7.0.SNAPSHOT)
[  67] [Active     ] [            ] [   80] Howlab::Zigbee Telegesis::Driver API (1.7.0.SNAPSHOT)
[  68] [Active     ] [            ] [   80] Howlab::Zigbee Telegesis::Driver Implementation (1.7.0.SNAPSHOT)
[ 149] [Active     ] [            ] [   80] Howlab::Serial Layer::Implementation OSGi Bundle (1.8.0)
[ 150] [Resolved   ] [            ] [   80] Howlab::RX-TX lib wrapper (1.2.0)
[ 185] [Installed  ] [            ] [   80] Coordinador (1.0.0.SNAPSHOT)
[ 188] [Installed  ] [            ] [   80] x73APIManager OSGi Bundle (1.0.0.SNAPSHOT)
```

Figura 6.1- Bundles utilizados en cada instancia.

El primer paso es abrir los puertos de los dispositivos ZigBee en cada una de las instancias que hemos creado. Una vez abierto el puerto pasamos a iniciar el bundle encargado de gestionar el manager (correspondiente al 185 en la figura 6.1), éste de un modo automático además de abrir la interfaz gráfica, pasará a la configurar lo registros de acuerdo a lo expuesto en el punto 4.1.1.1 y a la formación de la red ZigBee, una vez creada la red comienza a escanearla de un modo periódico, analizando todos los nodos que entren en ella, en el caso de que alguno disponga del perfil médico ZHC en alguno de sus perfiles, pasa a añadirlo en una lista de posibles agentes y a gestionarlo en función del tipo de dispositivo que se trata, en este caso lo reconoce como un termómetro.

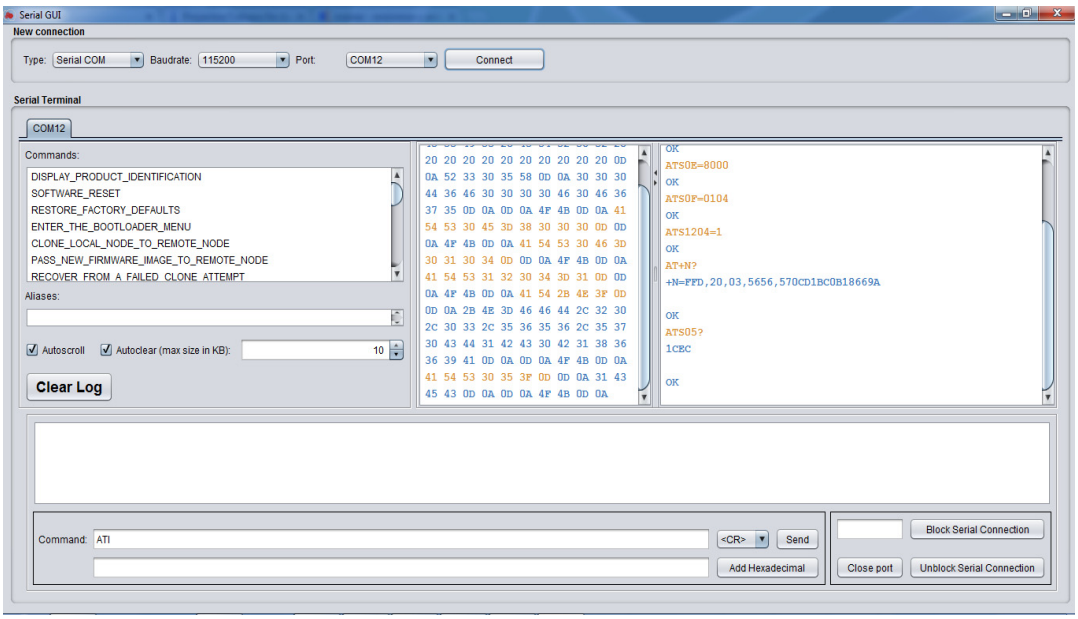


Figura 6.2 Interfaz del Serial GUI

Del mismo modo es necesario iniciar el sistema que gestiona el funcionamiento del Agente (166 en la figura 6.1), el cual tiene una actividad paralela al coordinador, comienza con configurar los registros que sean necesarios y a establecer la conexión con el sensor para formar el agente, una vez configurado comienza a escanear la red para encontrar al coordinador de esta que será el que implemente el manager, cuando lo encuentra establece un sistema para poder tratar los mensajes a nivel ZigBee. Llegados a este punto ya existe comunicación directa entre el router y el coordinador, además de con el sensor para implementar el agente.

El primer paso una vez establecida la comunicación a nivel de ZigBee es formar el canal ZHC, para ello comienza a enviar peticiones de conexión de un modo periódico hasta que como se indica en el apartado 4.1.2, recibe el permiso por parte del agente para iniciar el canal ZHC. En la figura 6.3 se puede comprobar cómo se completa este proceso, y como el mensaje es recibido en el endpoint 2 que a su vez implementa el perfil ZHC y dentro de éste al cluster 0614 que es el encargado de gestionar el protocol tunnel.

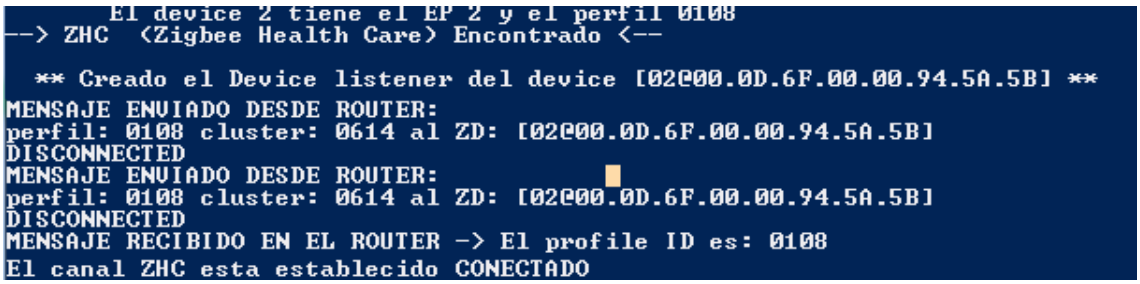


Figura 6.3 Establecimiento del canal ZHC

Una vez que el agente es consciente de que el canal ZHC se ha establecido, comienza a gestionar sobre éste el protocolo 11073 [1] a través del envío de tramas APDU. En primer lugar el dispositivo encargado de emular al agente envía una solicitud de asociación (figura 6.4), que será estudiada por el manager y contestará con un mensaje del tipo Association Response, el cual puede tomar valores del 0 al 8 cada uno de ellos con un significado diferente como se muestra a continuación:

- 0: Association Request Accepted
- 1: Association Request Rejected Permanent
- 2: Association Request Rejected Transient
- 3: Association Request Accepted Unknown Config
- 4: Association Request Rejected No Common Protocol
- 5: Association Request Rejected No Common Parameter
- 6: Association Request Rejected Unknown
- 7: Association Request Rejected Unauthorized
- 8: Association Request Unsupported Association Version

Este mensaje se puede observar en la figura 6.5 donde vemos como el resultado a dicha solicitud de asociación toma el valor 3 por lo tanto resulta una solicitud aceptada pero está a la espera de conocer la configuración del dispositivo. Así pues de un modo automático como se observa en la figura 6.4, el agente prepara su configuración en un Config Report o un informe de configuración y se la envía al manager a través de una trama APDU. El manager en este momento se encuentra en el estado *Esperando Configuración* como se puede observar en la interfaz gráfica.

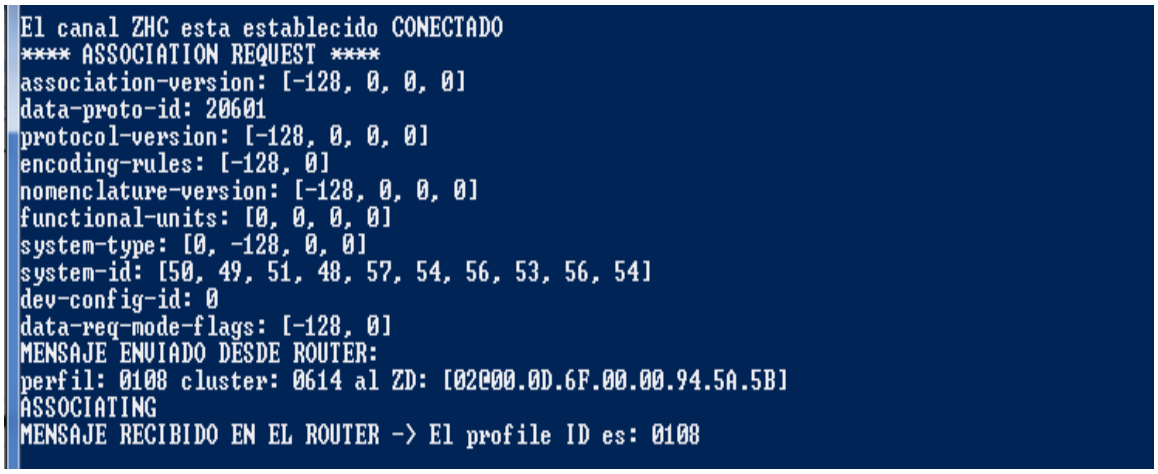


Figura 6.4- Solicitud de asociación

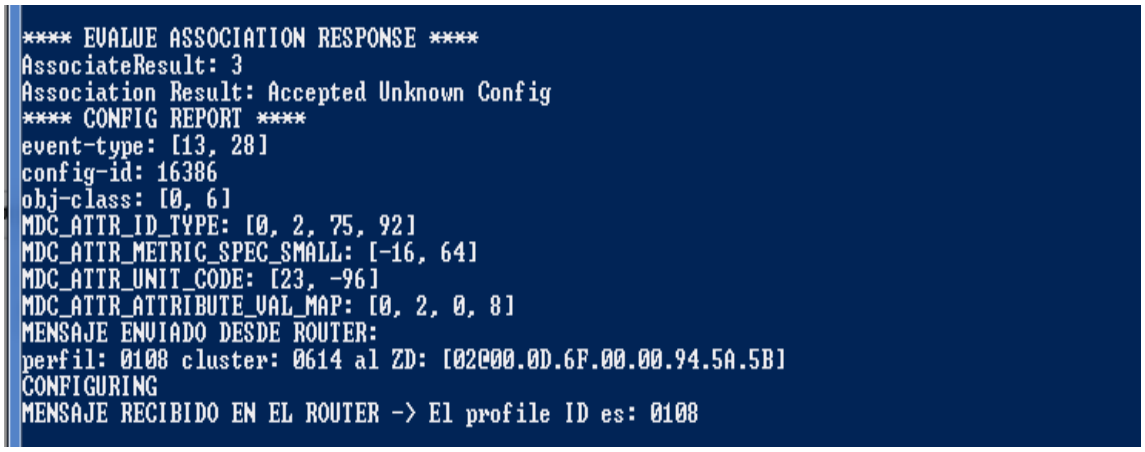


Figura 6.5 Evaluación de la solicitud de asociación

Una vez que esta configuración es recibida por el coordinador, esta pasa a evaluarla y elabora una respuesta a dicha configuración que puede responder de tres modos diferentes, con una respuesta satisfactoria(Config Response OK), con una aviso de que no soporta dicha configuración (Unsupported Config) o bien que ha aplicado una configuración estandar (Stándar Config Unknow).

Como vemos en la figura 6.6 la respuesta que da nuestro manager a la configuración del agente, es una respuesta satisfactoria, por lo que seguidamente ambos extremos alcanzan el estado operando.

Figura 6.6 Respuesta a la configuración

A partir de este momento, ambos extremos tanto el coordinador como el

manager se encuentran en el estado operando y es aquí cuando comienza todo el proceso de composición de la trama APDU, codificándola y siguiendo la estructura que se ha presentado en la figura 5.11. Este dato es enviado como se muestra en la figura 6.6 y enviara un mensaje de este tipo cada vez que tome una nueva medida. Este proceso se mantendrá hasta que de un modo voluntario o involuntario se proceda a la desconexión del sistema, pasando ambos extremos del canal al estado desconectado y posteriormente desconectando el canal ZHC.

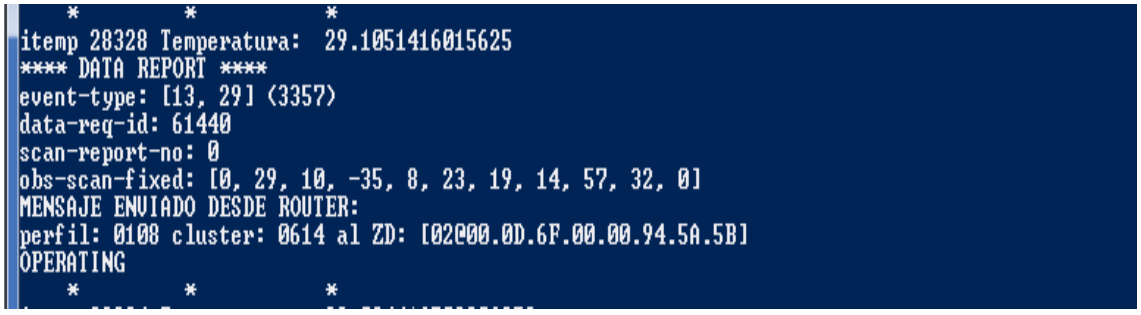


Figura 6.7 Dato de temperatura

En el manager se desarrolla el sistema de un modo paralelo al agente y para comprobar que todo funciona de un modo correcto nos apoyaremos en la interfaz gráfica que hemos diseñado (figura 6.8). El comportamiento por tanto es el siguiente, en primer lugar al iniciar el programa y conectar el dispositivo ETRX2 crea la red ZigBee de un modo inmediato y a continuación comienza a escanear la red para buscar los nodos, una vez que ha localizado el dispositivo que implementa el termómetro se mantiene a la espera de que este inicie la comunicación. Cuando acepta petición de crear el canal, comienza a actuar la máquina de estados del protocolo X73 donde en primer lugar pasa del estado desconectado al estado asociado, posteriormente se queda a la espera de recibir la configuración del agente para acabar alcanzando el estado operando cuando la recibe correctamente. En este estado es donde se produce el intercambio de tramas APDU con el dato de temperatura en su interior comprobamos que la decodificación se efectúa del modo correcto porque podemos observar el dato por pantalla junto con la hora a la que se ha tomado, además el propio manager almacenará todos los datos medidos para poder consultarnos cuando queramos.

Estos resultados fueron tomados trabajando en dos puntos de acceso diferentes en un mismo ordenador, del modo que se explica en el apartado 5.1. No obstante también se realizaron pruebas instalando la parte correspondiente al software que implementa el agente en otro PC y como cabía esperar se obtuvieron los mismos resultados.

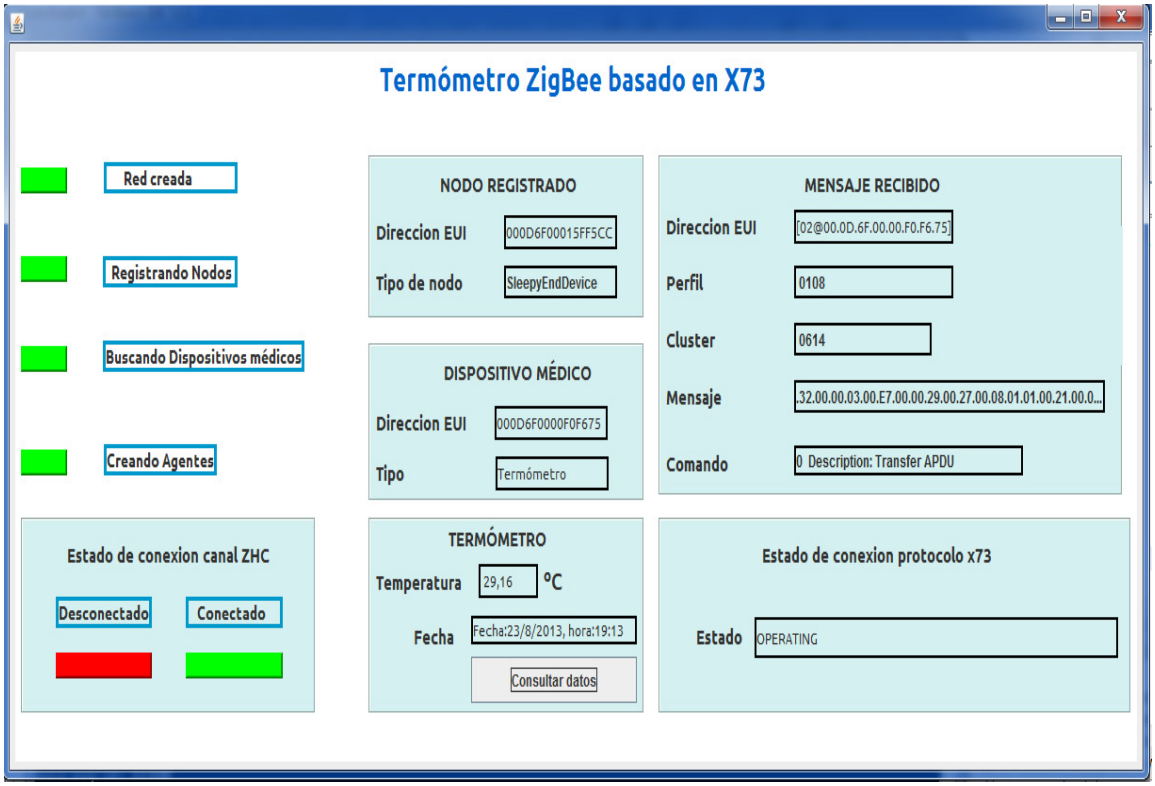


Figura 6.8 Interfaz gráfica en estado operando

7 Conclusiones y líneas futuras

En este apartado se muestran las líneas de trabajo que abre este proyecto fin de carrera y las posibles líneas de investigación a seguir una vez finalizado. Además de un estudio de las conclusiones obtenidas una vez finalizado el PFC.

7.1 Conclusiones

Una vez finalizado el PFC, llega el momento de realizar el balance del trabajo realizado, se puede afirmar por tanto que se han cumplido los objetivos marcados al inicio del proyecto donde se proponía desarrollar un sistema que conectase un manager y un agente médico por medio de tecnología ZigBee y de acuerdo al estándar 11073.

Por lo tanto se ha conseguido desarrollar una lógica capaz de gestionar de un modo automático todo el proceso necesario para establecer la comunicación entre ambos extremos. El sistema desarrollado cubre la gestión y creación de la red ZigBee con la correspondiente configuración de los dispositivos para que estos sean capaces de implementar el perfil público ZHC y que éste finalmente pueda actuar como soporte para el protocolo 11073.

En la parte correspondiente a la tecnología ZigBee trata desde la creación de la red, hasta el análisis de todos los nodos que se unen a ella estableciendo la comunicación entre el agente y el manager. Además los clusters que componen el perfil ZHC son capaces de tramitar los mensajes que reciben y el protocolo 11073 implementa todos los estados a ambos lados de la comunicación, con el correspondiente intercambio de tramas para acabar enviando el dato clínico de un modo satisfactorio, respetando de este modo todos los estándares y las normas necesarias para poder trabajar en un entorno médico real.

Por lo tanto se ha desarrollado por un lado toda la parte correspondiente al comportamiento de manager acompañado de una primera versión de la interfaz gráfica para comprobar tanto el dato clínico como el estado de la comunicación y por otro lado se ha diseñado la arquitectura necesaria para poder implementar un sensor médico capaz de trabajar en un entorno real.

Así pues una vez finalizado el proyecto queda perfectamente definido tanto el desarrollo a seguir en lo referente a la tecnología ZigBee, como la configuración necesaria de los módulos, dejando de este modo definida una línea de trabajo y una arquitectura a partir de la cual se puede seguir trabajando y puede resultar una perfecta base para proyectos futuros.

Por otro lado, el hecho de haber trabajado en un entorno de código abierto, nos permite poner todo el trabajo realizado a disposición de todo aquel que le pueda resultar de utilidad.

En este punto también es necesario indicar que en un principio el objetivo inicial era, una vez desarrollada toda la lógica que gestiona el agente trasladarla al hardware del sensor, para que este por si solo actúe como un agente médico. Pero finalmente por motivos de tiempo y carga de trabajo se decidió centrar el proyecto en desarrollar toda la arquitectura necesaria para implementar el agente, dejando la programación en el sensor para una segunda versión del proyecto.

Finalmente en un plano más personal, este PFC ha resultado muy

constructivo ya que me ha ofrecido la oportunidad de continuar con mi formación, familiarizándome con un nuevo entorno de trabajo y aprendiendo a desenvolverme en un entorno Java, que se trata de uno de los lenguajes más comunes en el mundo de la programación. Este proyecto también me ha ofrecido la oportunidad de estudiar y trabajar con una nueva tecnología, que prácticamente era desconocida para mi hace un año, como es la tecnología ZigBee y además implementar sobre ella un protocolo de comunicación como es el protocolo IEEE 11073.

Por último destacar que ha resultado una motivación extra el hecho de diseñar una plataforma de trabajo que pueda ser utilizada como base para futuros proyectos en un campo tan interesante como es la telemedicina obteniendo finalmente un desarrollo tangible y de utilidad.

7.2 Líneas Futuras

Una vez finalizado el proyecto son muchas las líneas de trabajo que quedan abiertas. El primer paso a tomar, es poner la plataforma en funcionamiento dentro de un entorno real para comprobar que todo se ha desarrollado correctamente. Para ello es necesario obtener un dispositivo médico que trabaje con tecnología ZigBee, soporte el protocolo 11073 y además estén certificados por Continua Alliance. Para adquirir estos dispositivos reales será necesario ponernos en contacto con la universidad de Brunnel, que como se explica en el estado del arte han conseguido la certificación de algunos de sus dispositivos o bien en la página acute technology [38], donde nos presenta un catálogo de diferentes dispositivos que soportan la norma X73 y el perfil ZHC.

Por otro lado, otra de las líneas de trabajo que se van a tomar se da en la parte correspondiente al sensor de temperatura, ya que una vez desarrollada toda la arquitectura a seguir, sería conveniente trasladar el software directamente sobre el hardware del sensor para que éste actúe por si solo como un agente médico, de este modo se podría optar a obtener en un futuro la homologación del producto por parte de Continua Health Alliance.

Una vez comprobado que el sistema funciona correctamente en un entorno real, se pasará a realizar los ajustes necesarios para que además de trabajar con un sensor de temperatura pueda soportar otros dispositivos médicos como puede ser un glucómetro o un tensiómetro entre otros, ya que esa es la finalidad que se busca para este proyecto, el hecho de que pueda trabajar en un entorno real con varios dispositivos.

Además gracias a la modularización de los bloques que componen el software que gestiona el proyecto siempre está sujeto a nuevos cambios y actualizaciones.

Así pues con la finalización de este PFC queda definida una estructura central capaz de operar en un entorno médico, sobre la cual se pueden desarrollar un amplio número de aplicaciones.

8 Bibliografía

[1] Estándar del protocolo 1107 Health informatics — Personal health device communication — Part 20601:Application profile Optimized exchange protocol
http://standards.ieee.org/findstds/standard/11073-20601-2008.html

[2] Howlab
http://howlab.unizar.es/

[3] Goodday Solutions S.L.
http://www.goodday.es/

[4] X73Spain Group
http://www.x73spain.com

[5] Telegesis
http://www.telegesis.com/

[6] ZigBee Alliance
http://www.zigbee.org/

[7] Estándar ZigBee Health Care
http://www.zigbee.org/Standards/ZigBeeHealthCare/Overview.aspx

[8] ZigBee Cluster Library Specification (ZCL)
ZigBee Document 075123r04ZB, May 29, 2012

[9] R. Wooton, J. Craig, “Introduction to Telemedicine”
Book Distributors, 2nd Edition, 2006.

[10] Fernando Barciela “*La telemedicina ya está aquí”*
Articulo publicado en EL PAIS, agosto de 2012
http://economia.elpais.com/economia/2012/08/24/actualidad/1345829355_158913.html

[11] Comunicado publicado en europapress,5 de abril de 2010
“El recientemente aprobado ZigBee Health Care Profile está disponible para descarga”
http://www.europapress.es/economia/noticia-comunicado-recientemente-aprobado-zigbee-health-care-profile-disponible-descarga-20100405150337.html

[12] American Recovery and Reinvestment Act de 2009
http://www.recovery.gov/About/Pages/The_Act.aspx

[13] Ignacio Martínez Ruiz, ¿el hospital en casa? dispositivos médicos personales

[14] HL7
http://www.hl7.org/

[15] Continua Health Alliance
http://www.zigbee.org/

[16] ZigBee Alliance
http://www.zigbee.org/

[17] Estándar IEEE 802.15.4
Ieee802.org. Retrieved 2012-10-18.
http://www.ieee802.org/15/pub/TG4.html

[18] Comunicado conjunto de ZigBee Alliance y The American Telemedicine *"The ZigBee alliance and the american telemedicine association to collaborate on advancing use of telehealth solutions"*

[19] Documento de ZigBee Alliance
"*ZigBee seleccionada por continua health alliance para las normas directivas de la próxima generación "*

[20] Tesis doctoral Javier Escayola Calvo,
"Contribuciones a estándares y tecnologías de comunicación en dispositivos médicos para e-Salud"

[21] Victor Kwong, Agosto 2011
"ZigBee Health Care Aplication Development Kit"
Freescale technology forum.

[22] Sergio R.Caprile, Editores GAE-2009
"Equisbí: Desarrollo de aplicaciones con comunicación remota basadas en ZigBee y 802.15.4"

[23] P. Del Valle García, J.D. Trigo Vilaseca,I. Martínez Ruiz, J. Escayola Calvo,M. Martínez-Espronceda Cámara, L. Serrano Arriezu, J. García Moros *"Interoperabilidad de dispositivos médicos mediante el estándar ISO/IEEE 11073 sobre tecnología Bluetooth".*

[24] D.Tejada, R.Achig, J.Fernandez, I. Martinez, M.Galárraga, L. Serrano, P. de Toledo"*Evolución de la interoperabilidad de dispositivos médicos y adaptación a la norma x73: casos de uso y diseño de implementación*"

[25] ETRX2USB ZigBee™ Wireless Mesh Networking USB Stick (Especificaciones del dispositivo ETRX2-USB) http://www.telegesis.com/downloads/general/ETRX2USB%20Product%20Brief.pdf

[26] ETRX2USB and ETRX2USB-PA USB STICK PRODUCT MANUAL (Manual del dispositivo ETRX2-USB) http://www.telegesis.com/downloads/general/TG-ETRX2USB-PM-004-105.pdf

[27] ETRX2 and ETRX3 Series ZigBee Modules AT-Command Dictionary http://www.telegesis.com/downloads/general/TG-ETRXn-R305-Commands.pdf

[28] Proyecto ZMota Ambi, (Especificaciones del sensor de temperatura) http://openlab.unizar.es/index.php/projects/zeta/zetas/ambi

[29] HOWlab: Zeta Project http://openlab.unizar.es/site/es.unizar.howlab.core.io.serial/faq.html

[30] ETRX2 and ETRX35x ZigBee MODULES Application Note-R3xx Interoperability, Telegesis

[31] Martínez, J. Escayola, I. Fernández de Bobadilla,M. Martínez-Espronceda, L. Serrano, J. Trigo, S. Led, y J. García *"Optimización de una Plataforma Telemática para Monitorización de Pacientes orientada a u-Salud, y basada en Estándares y Plug-and-Play"*

[32] Pedro Funes Salas Proyecto fin de carrera , Julio 2010 *"Implementación de una plataforma de telemonitorización de pacientes estándar, open source y ubicua basada en ISO/IEEE 11073-PHD sobre Android."*

[33] Osgi alliance http://www.osgi.org

[34] Apache Felix http://felix.apache.org/

[35] Karaf http://karaf.apache.org/

[36] Openlab-Project Zeta- Device Zeta Kit http://openlab.unizar.es/index.php/projects/zeta/software/zeta-devkit

[37] Apache MAVEN http://maven.apache.org/

[38] Acute technology
http://www.acutetechnology.com/

[39] THK SENSOR
http://openlab.unizar.es/site/es.unizar.howlab.projects.renaissance/renaissance-thk-sensor/

[40] RENAISSANCE Project
http://www.renaissance-project.eu/?lang=es

[41] Programa de Postgrado en Ingenierías Transversales Ingeniería Biomédica · Mención de Calidad. Autor Antonio Aragüés Ruiz Director Dr. Ignacio Martínez Ruiz. Diciembre 2010.
"Contribuciones en el diseño, implementación e integración de nuevas tecnologías y perfiles médicos recomendados para entornos de e-Salud basados en el estándar ISO/IEEE 11073"

A1 - Anexo 1: Cronograma de implantación

En este apartado se presenta un seguimiento temporal de cómo se ha elaborado el PFC, quedando resumido gráficamente en un diagrama de Gantt en el apartado dos.

Anexo 1.1 Cronograma de implementación

La duración del proyecto se divide más o menos a partes iguales entre la parte correspondiente a la tecnología ZigBee y la implantación del protocolo 11073, la primera parte se llevó a cabo en los laboratorios del grupo de investigación Howlab, desde octubre de 2011 hasta finales de febrero y la parte final en las oficinas de la empresa Goodday Solutions.

En primer lugar se realizó un trabajo de investigación acerca de la tecnología ZigBee y todo el entorno de trabajo que se iba a utilizar a lo largo del proyecto.

Para comenzar a familiarizarnos con el entorno se realizaron una serie de tutoriales de la página de Apache Felix [18], para comprender el contexto Maven y el lenguaje Java en general.

El siguiente paso fue comenzar a realizar algunas pruebas a nivel de Zigbee como puede ser la formación de la red, o el envío de mensajes entre dispositivos, en primer lugar de un modo manual y posteriormente poco a poco se comenzó a elaborar el software que lo gestionase de un modo automático. Además se realizaron pruebas con el sensor de temperatura como se explica en el Anexo 4 y para ello se tomó como referencia un subproyecto desarrollado por el grupo Howlab con el nombre de THK-Sensor [31], que a su vez es otro subproyecto del proyecto Reinassance [32.]

Posteriormente se paso a realizar un estudio detallado de todos los registros que componen los módulos de Telegesis, para poder finalmente alcanzar la configuración presentada en el apartado 4.1.1 (Anexo 2).

Una vez que se consiguió que la formación de la red y el análisis de los dispositivos que la componían de un modo automático por medio de dos bundles, uno encargado de gestionar el coordinador y otro encargado de la gestión del router, se pasó al estudio del perfil público ZHC que nos proporciona Zigbee Alliance para la gestión de dispositivos médicos, para en primer lugar configurar el dispositivo para poder soportar dicho perfil en uno de los endpoint del dispositivo ZigBee y en segundo desarrollar el software capaz de crear un canal entre un manager y un agente de un modo automático como se indica en el apartado 4.1.2.

El último paso fue toda la parte correspondiente a la implantación de la norma X73 sobre nuestro sistema, se paso a trabajar del mismo modo que con las partes anteriores un primer periodo de aprendizaje hasta que se pudo pasar trabjar con el protocolo. Para finalmente implementarlo en el sistema fue necesario adaptar unos códigos facilitados por Goodday que nos daban las herramientas necesarias para implementar la máquina de estados capaz de gestionar las fases del protocolo 11073.

Los últimos meses se utilizaron para desarrollar la interfaz gráfica y realizar las pruebas necesarias para comprobar el correcto funcionamiento del sistema.

Anexo 1.2 Diagrama de Gantt

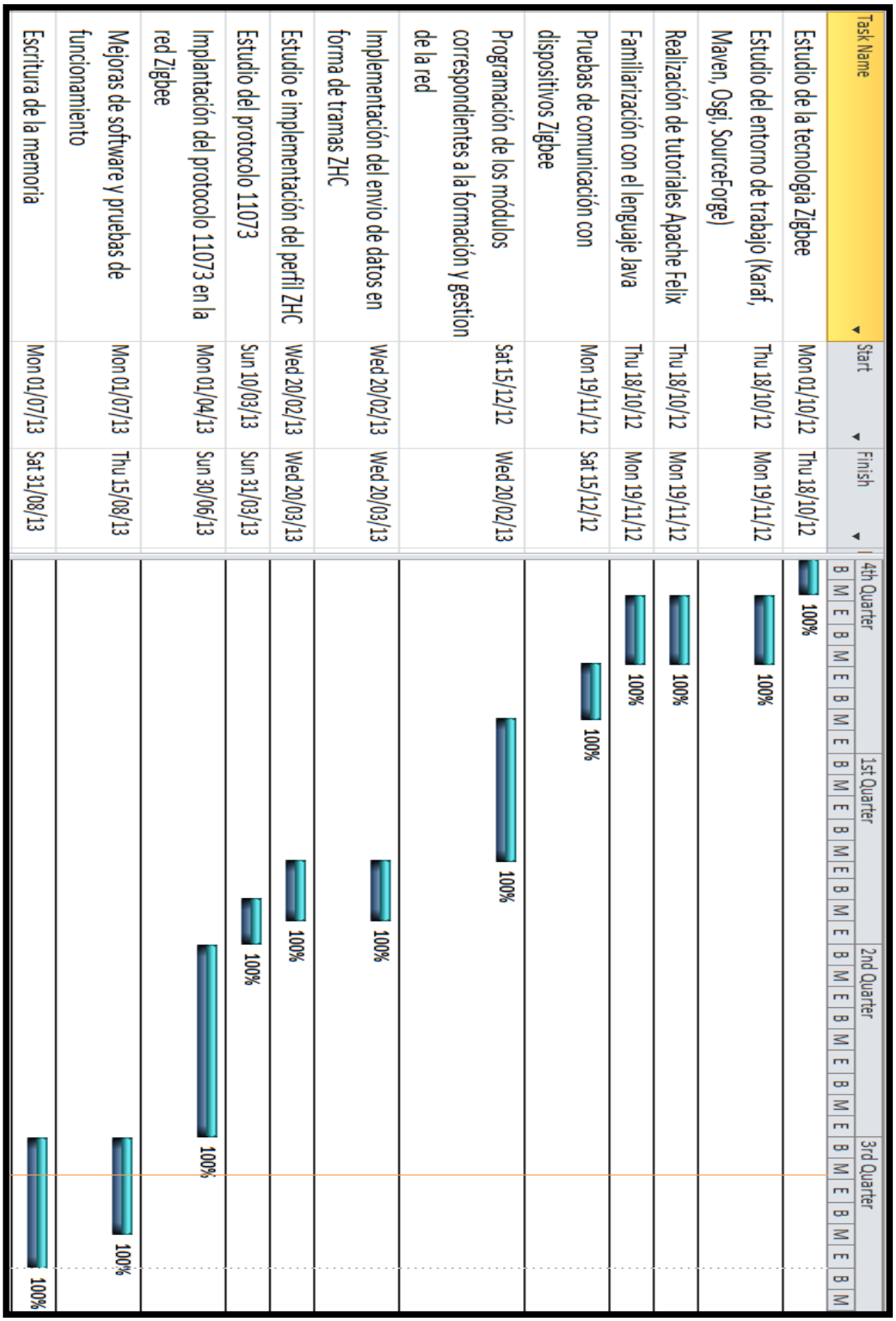


Figura A1.1- Diagrama de Gant

A2 - Anexo 2: Configuración de registros

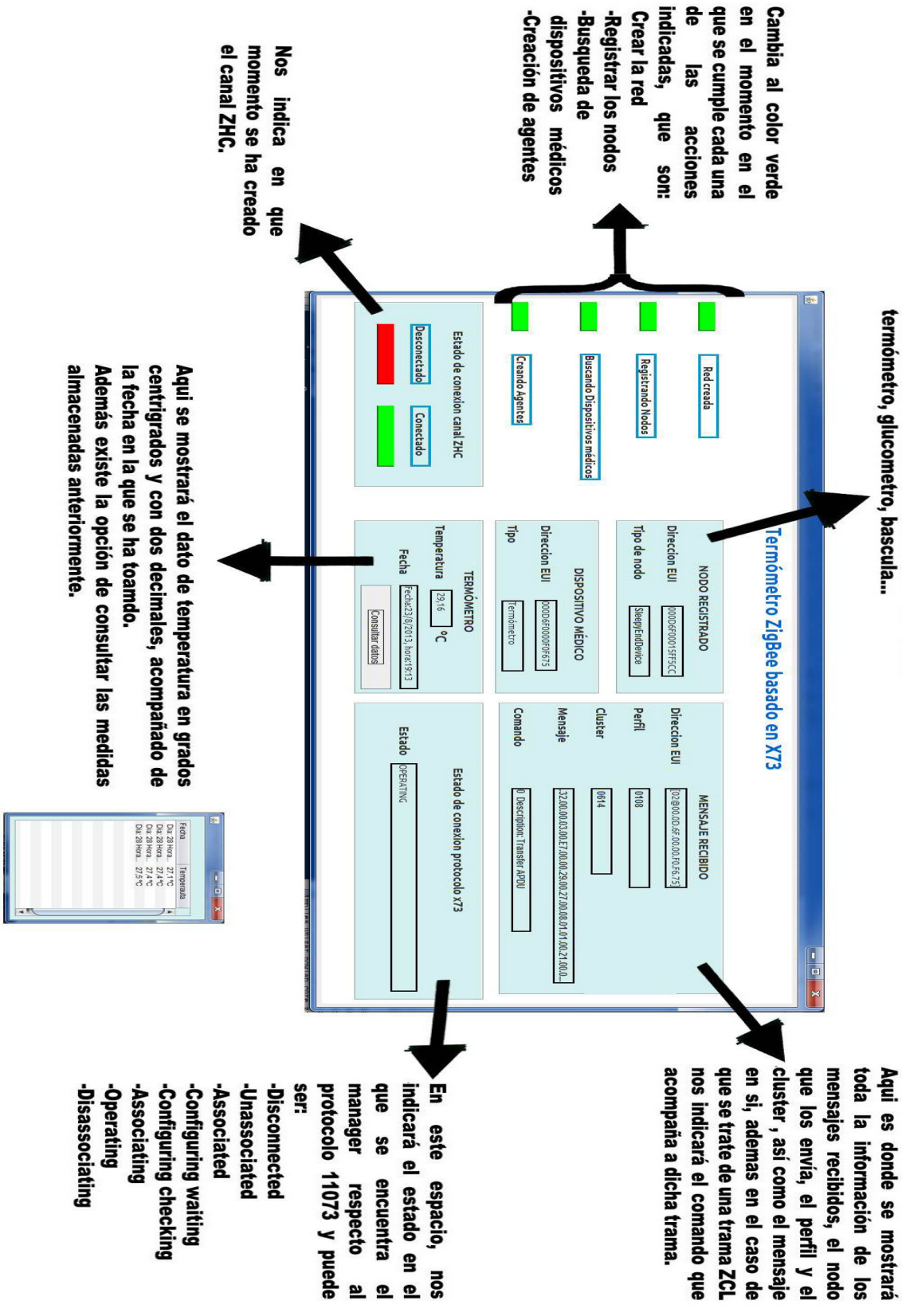
Registro	Descripción	Valor COO	Valor FFD
S00	Channel Mask	0200	FFFF
S01	Transmit Power Level	01	1
S02	Preferred PAN ID*	5656	5656
S03	Prefered Extended PAN ID	0000000000000000	0000000000000000
S04	Local EUI	000D6F0000945A5B	000D6F0000F0F675
S05	Local NodeID	0000	C44C
S06	Parent´s EUI	0000000000000000	0000000000000000
S07	Parent´s NODEID	FFFF	FFFF
S08	Network Key*	default	default
S09	Link Key*	default	default
S0A	Main Function	181C	0D14
S0B	User Readable Name	Telegesis	Telegesis
S0C	Password	password	password
S0D	Device Information	ETRX2 R305X	ETRX2 R305X
S0E	Promt Enable 1*	8000	8000
S0F	Promt Enable 2 *	0104	0104
S10	Extended Function	159A	14CA
S11	Device Specific	0300	0300
S12	UART Setup	0C10	0C10
S13	Pull-up enable	0000	0
S14	Pull-down enable	0000	0000
S15	I/O Configuration	0000	0000
S16	Data Direction of I/O Port	00F8	00F8
S17	Initial Value of S16	00F8	00F8
S18	Output Buffer of I/O port	00F0	00F0
S19	Initial Value of S18	00F0	00F0
S1A	Input Buffer of I/O port	0FF7	0FF7
S1B	Special Function pin 1	3A98	3A98
S1C	Initial Value of S1B	3A98	3A98
S1D	Special Function pin 2	1D4C	1D4C
S1E	Initial Value of S1D	1D4C	1D4C
S1F	A/D1 (ETRX3: ADC0)	0262	025D
S20	A/D2 (ETRX3: ADC1)	262	025E
S21	A/D3 (ETRX3: ADC2)	0001	1
S22	A/D4 (ETRX3: ADC3)	0001	1
S23	Immediate functionality at IRQ0	0001	1
S24	Immediate functionality at IRQ1	0000	0
S25	Immediate functionality at IRQ2	0000	0000
S26	Immediate functionality at IRQ3	0000	0000
S27	Functionally 1 at Boot-up	0000	0000

Tabla A2. 1- Registros S00- S27

Registro	Descripción	Valor COO	Valor FFD
S28	Functionality at Network Join	0000	0000
S29	Timer/Counter 0	4	4
S2A	Functionality for Timer/Counter 0	8010	8010
S2B	Timer/Counter 1	00F0	00F0
S2C	Functionality for Timer/Counter 1	821E	821E
S2D	Timer/Counter 2	00F4	00F4
S2E	Functionality for Timer/Counter 2	8014	8014
S2F	Timer/Counter 3	00F2	00F2
S30	Functionality for Timer/Counter 3	8015	8015
S31	Timer/Counter 4	0000	0
S32	Functionality for Timer/Counter 4	0000	0
S33	Timer/Counter 5	0000	0
S34	Functionality for Timer/Counter 5	0000	0
S35	Timer/Counter 6	0000	0
S36	Functionality for Timer/Counter 6	0000	0
S37	Timer/Counter 7	0000	0
S38	Functionality for Timer/Counter 7	0000	0
S39	Power Mode(volatile)	0000	0000
S3A	Initial Power mode	0000	0
S3B	Start-up Funcitonally Plaintext A	BUTTON 3	BUTTON 3
S3C	Start-up Funcitonally Plaintext B	BUTTON 4	BUTTON 4
S3D	Supply Voltaje	3432	3344
S3E	Multicast Table Entry 00	0000	0
S3F	Multicast Table Entry 01	0000	0
S40	Source and Destination Endpoint for xCASTs*	202	202
S41	Initial Value of S40	0101	101
S42	Cluster ID for xCASTs (volatile)*	0000	0
S43	Initial value of S42	0000	0
S44	Profile ID for xCASTs (volatile)	0108	108
S45	Initial value of S44	C091	C091
S46	Start-up Functionally 32 bit numer (volatile)	0	0
S47	Power descriptor	C110	C110
S48	Endpoint 2 Profile ID*	0108	108
S49	Endpoint 2 Device ID*	0000	1008
S4A	Enpoint 2 Device versión*	0000	0
S4B	Enpoint 2 Input Cluster List*	0000 0003 0600 0614	614
S4C	Enpoint 2 Output Cluster List*	0614	0000 0003 0600 0614
S4D	Mobile End Device Poll Timeout	1000	14
S4E	End Device Poll Timeout	0EFF	605
S4F	MAC Timeout	F000	0BB8

Tabla A2.2 – Registros S28-S4F

A3 - Anexo 3: Interfaz gráfica



A4 - Anexo 4: Emulación del agente

Debido a la imposibilidad de poder trabajar con dispositivos médicos reales, dotados de tecnología ZigBee y que a su vez cumpliesen el estándar 11073, se decidió simular toda la parte correspondiente al agente.

La idea de diseño que se llevó a cabo para montar el agente fue en primer lugar montar toda la lógica correspondiente a la gestión del propio agente sobre uno de los módulos de Telegesis, más concretamente sobre el endpoint encargado de implementar el perfil médico ZHC, mientras que el dato de temperatura será suministrado a través del sensor presentado en el apartado 3.3.2 a través de una red ZigBee.

Por lo tanto, la comunicación entre el router y el sensor se realiza a través de una red ZigBee, por lo que fue necesario en primer lugar comprender el funcionamiento de dicha red, para ello, se estuvieron realizando varias pruebas tanto a nivel de software como de hardware, a través de una red formada por un coordinador, un router y el propio sensor, dentro de dicha red el sensor actúa como un Sleepy End Device, es decir solo se activa cuando es necesario que el dato sea transmitido permaneciendo la mayor parte del tiempo dormido, es aquí donde radica el secreto de que las baterías duren tanto tiempo.

Para llevar a cabo estas pruebas se tomó como referencia un subproyecto desarrollado por el grupo Howlab con el nombre deTHK-Sensor [31], que resulta a su vez un subproyecto del proyecto Reinassance [32.]. A partir de este trabajo se consiguieron las herramientas necesarias para controlar el sensor, tratando desde el propio dato de temperatura hasta posibles alertas que este pueda enviar al router como por ejemplo una alerta que se envía cuando el nivel de batería es muy bajo.

En la figura A4.1 se muestra como el router (conectado a través del puerto COM 12) recibe un mensaje por parte del sensor de temperatura (cuya dirección es 000D6F00015FF5CC) con el dato de temperatura, que tendrá que ser posteriormente decodificado como se indica en la figura A.4.2.

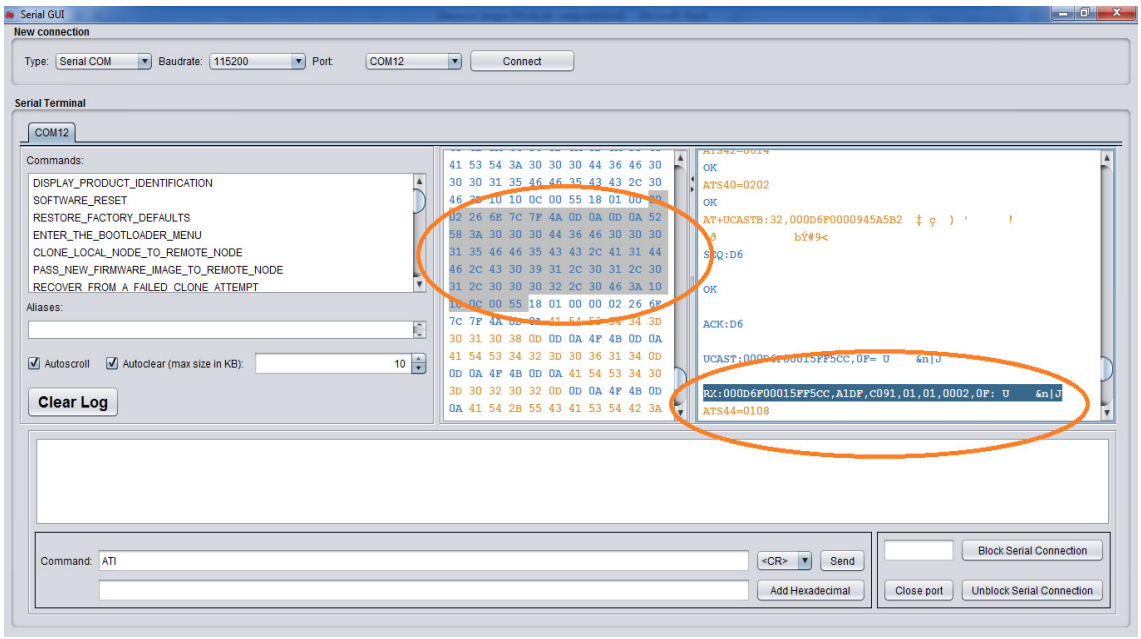


Figura A4.1 Interfaz del router recibiendo un mensaje del sensor

```
byte[] data;
short dataLength = (short) ((message[2] & 0xFF) - 1);
if (dataLength != (message.length - 4)) {
     data= null;
     } else if (dataLength == 0) {
      data= null;
     } else {
     data = new byte[dataLength];
     System.arraycopy(message, 4, data, 0, dataLength);
  }
            int iTemp = (short) (data[7] & 0xFF) << 8;
 iTemp += (short) (data[8] & 0xFF);
 double temp = (175.72 * (iTemp / (Math.pow(2, 16)))) - 46.85;
```

Figura A4.2 Código que obtiene el dato de temperatura

Así pues en cuanto el software encargado de gestionar el agente, nada más que detecte el driver pasará a establecer un sistema de escucha con el sensor y de este modo podremos trabajar con el dato de temperatura como si fuese el propio router el que lo generase, quedando este proceso completamente invisible para las capas superiores y tratando a partir de aquí a la combinación de los dos dispositivos como uno único.

La decisión de desarrollar un software que trabaje a partir de esta simulación de un agente en lugar de pasar directamente a programar sobre el sensor, se debe a que ya que la idea del proyecto es desarrollar una lógica capaz de gestionar todos los protocolos de la comunicación entre dos dispositivos médicos, resultaba más cómodo desarrollarla de un modo simultaneo al desarrollo del manager y una vez finalizado el proyecto ya estaríamos en disposición de trasladar todo el sistema de gestión sobre el sensor, para que este trabaje por sí solo.